

Aspect Workforce Cloud™ SDK Developer Guide

RELEASE DATE: 8/25/2023
REVISED DATE: 11/7/2025

Legal Notices

© 2025 Alvaria, Inc. Unauthorized reproduction prohibited by law.

The content of this publication is furnished for informational use only and should not be construed as a commitment by Alvaria, Inc. f/k/a Aspect Software, Inc. (“Alvaria”). Alvaria assumes no responsibility or liability for any errors or inaccuracies that may appear in this publication. Alvaria reserves the right to change information in this publication without notice as a result of product enhancements or other reasons.

Alvaria™, Aspect®, Unified IP® and other marks as indicated are trademarks or registered trademarks of Alvaria, Inc. in the United States and other countries. Use of any Alvaria trademark is prohibited unless expressly approved in writing in advance by an authorized representative of Alvaria, Inc. Microsoft Windows®, and Microsoft SQL Server® are registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Any other brands, product names, company names, logos, trademarks, and/or service marks used in this publication are the property of their respective owners. You may not copy, modify or display any of Alvaria’s or its affiliates’ or licensors’ trademarks, trade names or logos appearing in this publication in any way without Alvaria’s express written consent.

The works of authorship, including but not limited to all design, text and images, contained and the software described in this publication are owned by Alvaria or its affiliates or licensors, except as otherwise expressly stated. The entire contents of this publication are protected by United States and worldwide copyright laws and treaty provisions. In accordance with these laws and provisions, you may not copy, reproduce, modify, use, republish, upload, post, transmit or distribute in any way material from this publication. Further, except as permitted by your written agreement with Alvaria, no part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, or otherwise, without the prior written permission of Alvaria.

RESTRICTED RIGHTS LEGEND

This publication is provided with “Restricted Rights”. No part of this publication may be photocopied, reproduced or transmitted, in any form or by any means, without the prior written consent of Alvaria. Use, duplication, or disclosure by the United States Government (“Government”) is subject to the restrictions set forth in DFARS 252.227-7013 (b)(3) and FAR 52.227-19. Use of the materials by the Government constitutes acknowledgement of Alvaria’s proprietary rights in them. Alvaria is located at 211 Perimeter Center Parkway NE, Suite 250, Atlanta, GA 30346, USA.

LIMITED RIGHTS NOTICE (DEC 2007)

(a) These data are submitted with limited rights under Alvaria’s contracts with various Government entities. These data may be reproduced and used by the Government with the express limitation that they will not, without written permission of the Alvaria, be used for purposes of manufacture nor disclosed outside the Government; except that the Government may disclose these data outside the Government for the following purposes, if any, provided that the Government makes such disclosure subject to prohibition against further use and disclosure: None.

(b) This notice must be marked on any reproduction of these data, in whole or in-part.

EXPORT

This item is subject to U.S. export control laws and regulations. This item may not be exported, re-exported, re-transferred, disclosed or otherwise diverted contrary to U.S. export control laws or regulations.

NO WARRANTY

THE CONTENTS OF THIS PUBLICATION ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF QUALITY, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR THAT OPERATION OF THE PRODUCTS SOLD BY ALVARIA WILL BE UNINTERRUPTED OR ERROR FREE.

NO LIABILITY

ALVARIA, ITS AFFILIATES, AND LICENSORS ARE NOT LIABLE FOR ANY DAMAGES SUFFERED AS A RESULT OF USING THE CONTENTS OF THIS PUBLICATION. IN NO EVENT WILL ALVARIA, ITS AFFILIATES OR LICENSORS BE LIABLE FOR ANY (i) CONSEQUENTIAL, INDIRECT, PUNITIVE, SPECIAL, OR INCIDENTAL DAMAGES, (ii) ANY INTERRUPTION OF BUSINESS OR OPERATIONS, COST OF COVER, GOODWILL, TOLL FRAUD, OR LOSS OF DATA, PROFITS, OR REVENUE, OR (iii) FAILURE OF A REMEDY TO ACHIEVE ITS ESSENTIAL PURPOSE. THE LIMITATIONS IN THIS SECTION WILL APPLY TO ANY DAMAGES, HOWEVER CAUSED, AND ON ANY THEORY OF LIABLITY, WHETHER FOR BREACH OF CONTRACT, TORT, MISREPRESENTATION, NEGLIGENCE, THE USE OR PERFORMANCE OF A PRODUCT OR SERVICE, OR OTHERWISE AND REGARDLESS OF WHETHER THE DAMAGES WERE FORESEEABLE OR UNFORSEEABLE. NEITHER PARTY WILL BE LIABLE FOR ANY CLAIM BROUGHT BY THE OTHER PARTY MORE THAN 12 MONTHS AFTER THE OTHER PARTY BECAME AWARE OF THE ISSUE GIVING RISE TO THE CLAIM. ALVARIA’S, ITS AFFILIATES’ OR LICENSORS’ FAILURE TO EXERCISE A RIGHT OR REMEDY IS NOT A WAIVER. BECAUSE SOME JURISDICTIONS PROHIBIT THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

PROGRAMMING AND USE OF PRODUCTS

THE PRODUCTS DESCRIBED IN THIS PUBLICATION CAN BE USED AND PROGRAMMED IN A WIDE VARIETY OF WAYS BASED ON THE REQUIREMENTS OF YOUR PARTICULAR TECHNOLOGY, ENVIRONMENT AND BUSINESS. NOTWITHSTANDING THE USE OF EXAMPLES IN THIS PUBLICATION OR THE PROVISION OF PROFESSIONAL SERVICES BY ALVARIA, ALVARIA RESELLERS, OR ANY THIRD PARTY ENGAGED BY ALVARIA, IT IS IN ALL CASES YOUR RESPONSIBILITY TO ENSURE THAT THE PRODUCTS ARE PROGRAMMED AND USED IN ACCORDANCE WITH ALL APPLICABLE LAWS AND REGULATIONS AND IN A MANNER THAT DOES NOT VIOLATE THE INTELLECTUAL PROPERTY OR OTHER RIGHTS OF ANY THIRD-PARTY.

Rev D
November 7, 2025

Contents

Contents **3**

Revision History **21**

About this Guide **22**

Audience 22

Organization of this Guide 22

About Backward Compatibility 24

Installing the SDK **25**

API Guides **27**

Tally Server API Developer Guide 27

Related Software Files 27

File List (Partial) 27

Prerequisites 28

Additional Requirements 29

Audience 29

Additional Documentation 29

Overview 29

Tally Server Communications 30

Aspect Workforce Client 30

Real-Time Adherence Service 31

Administrators 31

API Goals and Assumptions 31

Tally Server Request Types 31

Installing the API 33

Concepts in Aspect Workforce 34

Surrogate Keys (SK) 34

Perspectives, States, Segment Codes, Superstates 34

Perspectives 34

Perspective of Basic Shift Segment 35

Perspectives of Call Handling Segment 35

Perspectives and Ranking ... 36
Perspectives from the Top-Down View ... 36
States ... 37
Schedules and Segments ... 37
Segment Codes ... 37
Segment Categories ... 38
Segment Entry Rules ... 38
Segment Types ... 38
Schedule Segments ... 38
Official Schedules ... 39
Trial Schedules ... 39
Superstates ... 39
UTC Time ... 41
Moments ... 41
Fine Moments ... 41
Nominal Dates ... 42
Employee Groups ... 42
Staff Groups ... 42
Data Types and Stored Procedures ... 43
About Data Types ... 43
About Stored Procedures and Surrogate Keys ... 44
GET_SK ... 44
GET_SKS ... 44
Database Tables ... 45
Official Segment Tables ... 45
EMP ... 45
GEN_SEG ... 46
DET_SEG ... 46
DLY_SEG_SMRY ... 47
Segment Information Tables ... 47
SEG_CODE ... 47
SEG_CAT ... 48
SEG_CODE_CAT_ASSOC ... 48
Other Tables ... 49
PERSP ... 49
STATE ... 49
SEG_STATE_ASSOC ... 50

SPRSTATE_MBR 50
SPRSTATE 50
SPRSTATE_EXCL_DAYS 51
SPRSTATE_TIME_FLT 51
SPRSTATE_DUR_FLT 51
SPRSTATE_HRS 52
Tally Server Architecture 52
Tally Server Server 53
Tally Transport 53
Tally Server Concepts 53
Resolution Request 53
Segment Selector 54
Schedule and Resolution Identifier 55
Internal Processing 55
Tally Server API Implementation 56
Basic Programming Model 57
COM Interfaces Reference 57
About the ITCSTallyServer Tally Server Interface 57
ITCSTallyServer Interface 58
GetMomentBasedResolutionProvider Method 58
GetMomentBasedTimePeriodTallyProvider Method 59
GetNomDateBasedScheduleListTallyProvider Method 62
GetNomDateBasedScheduleTallyProvider Method 64
GetMomentBasedScheduleListTallyProvider Method 66
ITCSMomentBasedResolutionProvider Interface 68
GetResolvedScheduleInfo Method 68
ITCSMomentBasedTimePeriodTallyProvider Interface 70
GetTallyInfo Method 70
ITCSNomDateBasedScheduleListTallyProvider Interface 71
GetTallyInfo Method 71
ITCSNomDateBasedScheduleTallyProvider Interface 72
GetTallyInfo Method 72
ITCSMomentBasedScheduleListTallyProvider Interface 73
GetTallyInfo Method 73
Common Type and Property Definitions 74
Common Type Structure Definitions 74
Data Types of Common Properties 75

Sample Code and Required Files 76
Sample Code 76
Required Files 77
Updater API Developer Guide 80
Related Software Files 80
Prerequisites 80
Additional Requirements 80
Audience 81
Additional Documentation 81
Overview 81
Components of the Updater API 81
Installing the Updater API 82
Concepts in Aspect Workforce 82
Surrogate Keys (SK) 82
Schedules and Segments 82
UTC Time 82
Moments 83
Nominal Dates 83
Segment Codes 83
Segment Categories 83
Segment Entry Rules 83
Segment Types 83
General Segments 84
Detail Segments 84
Data Types and Stored Procedures 84
About Data Types 84
About Stored Procedures and Surrogate Keys 85
GET_SK 85
GET_SKS 86
Database Tables 86
Official Segment Tables 86
EMP 86
GEN_SEG 87
DET_SEG 87
DLY_SEG_SMRY 88
Segment Information Tables 88
SEG_CODE 88

SEG_CAT ... 89
SEG_CODE_CAT_ASSOC ... 89
SEG_CAT_RULE ... 90
Updater Architecture ... 90
Updater Server ... 91
Updater Client ... 91
Updater Transaction ... 92
Updater Validation Engine ... 92
Updater Validation: Intrinsic Rules ... 92
Updater Validation: User-Defined Rules ... 97
About the Updater API ... 98
Basic Programming Model ... 98
Updater Programming Model in Detail ... 98
Updater Transaction ... 99
Updater Actions ... 99
Updater Violations ... 100
Updater Errors ... 100
About the Visual Basic Wrapper ... 100
Updater Error Codes ... 100
COM Interfaces Reference ... 104
IUpdaterTransaction ... 104
Implementation ... 105
Methods (Listed) ... 105
Properties (Listed) ... 105
Methods (Documented) ... 106
Properties (Documented) ... 110
IUAGsegInsert ... 115
Implementation ... 115
Properties (Listed) ... 116
Properties (Documented) ... 116
IUAGsegUpdate ... 118
Implementation ... 118
Properties (Listed) ... 118
Properties (Documented) ... 119
IUAGsegDelete ... 122
Implementation ... 122
Properties (Listed) ... 122

Properties (Documented) 122
IUADsegInsert 124
Implementation 124
Properties (Listed) 124
Properties (Documented) 125
IUADsegUpdate 127
Implementation 127
Properties (Listed) 127
Properties (Documented) 128
IUADsegDelete 132
Implementation 132
Properties (Listed) 132
Properties (Documented) 133
IUpdaterViolation 135
Implementation 135
Properties (Documented) 136
Implementation 138
Properties (Listed) 138
Properties (Documented) 139
IUpdaterSuperstateSummaryFilter 139
Implementation 140
Methods (Listed) 140
Methods (Documented) 140
TBool 141
TUActionType 142
TUPriority 142
TUStatus 143
Other Notes 144
About the Updater Type Libraries 144
Programming with Other Languages 144
Using the Custom Interfaces 145
Visual Basic Reference 145
Locale Data Format 145
Sample Code 145
Updater Plug-In Rule Developer Guide 149
Related Software Files 149
Prerequisites 149

Audience 149
List of Features 150
Organization of this Guide 150
About Updater Plug-In Rules 150
Architecture 150
Rule Validation 151
Plug-In Rule Configuration Information 152
Workforce 8.2 and later and 64-bit 153
Implementation Steps 153
Deployment Steps 154
API 154
GetConfigurationInfo 154
DBAlias 154
ConfigString 155
Validate 155
ConfigString 155
Transaction 155
Interaction Diagram 157
Error Messages 158
Type Library 158
Performance Guidelines 158
Installing and Configuring Plug-In Rules 158
Installing the DLL 158
Configuring Updater for Plug-In Rules 159
Identifying the Plug-In Rule DLL 159
Invoking the Plug-In Rule 159
Configuring Plug-In Rules 160
Sample Program 161

Notification Server API Developer Guide 163
Related Software Files 163
Prerequisites 163
Audience 164
About Notification Server 164
About the Notification Server API 164
List of Features 164
List of Interfaces 164
List of Properties and Methods by Interface 165

IAsptContactInfoProvider Interface 165
IAsptContactInfo Interface 165
IAsptNotificationSubmissionProvider Interface 165
IAsptNotificationManagementProvider Interface 166
IAsptNotification Interface 166
IAsptNotificationAttachment Interface 167
IAsptNotificationDispatcher Interface 167
IAsptNotificationXMLGenerator Interface 167
IAsptNotificationStreamProvider Interface 167
IAsptNotificationStream Interface 167
Interface Definitions 168
IAsptContactInfoProvider 168
IAsptContactInfoProvider::UserName Property 168
IAsptContactInfoProvider::Password Property 168
IAsptContactInfoProvider::ConnectionName Property 169
IAsptContactInfoProvider::CLSID Property 169
IAsptContactInfoProvider::DBAlias Property 170
IAsptContactInfoProvider::GetContactInfo Method 170
IAsptContactInfoProvider::Open Method 171
IAsptContactInfoProvider::Close Method 171
IAsptContactInfo 172
IAsptContactInfo::ContactID Property 172
IAsptContactInfo::ContactProperty Property 172
IAsptNotificationSubmissionProvider 173
IAsptNotificationSubmissionProvider:: NotificationStatus Property 173
IAsptNotificationSubmissionProvider:: NumNotificationsHint Property 174
IAsptNotificationSubmissionProvider:: ConnectionName Property 175
IAsptNotificationSubmissionProvider:: CreateNotification Method 176
IAsptNotificationSubmissionProvider:: SubmitNotification Method 176
IAsptNotificationSubmissionProvider:: CreateAttachment Method 177
IAsptNotificationSubmissionProvider:: SubmitAttachment Method 177
IAsptNotificationSubmissionProvider::Open Method 178
IAsptNotificationSubmissionProvider::Close Method 178
IAsptNotificationManagementProvider 179
IAsptNotificationManagementProvider:: NumNewNotifications Property 179
IAsptNotificationManagementProvider::Notification Property 180
IAsptNotificationManagementProvider::NotificationStatus Property 180

IAsptNotificationManagementProvider::UserName Property 181
IAsptNotificationManagementProvider::Password Property 182
IAsptNotificationManagementProvider::ConnectionName Property 182
IAsptNotificationManagementProvider:: NumAttachments Property 183
IAsptNotificationManagementProvider::GetNewNotification Method 183
IAsptNotificationManagementProvider::GetNextAttachment Method 184
IAsptNotificationManagementProvider::Open Method 185
IAsptNotificationManagementProvider::Close Method 185
IAsptNotificationManagementProvider:: UpdateNotificationProcTs Method 186
IAsptNotificationManagementProvider:: UpdateNotificationDispatchID Method 186
IAsptNotificationManagementProvider:: UpdateNotificationStatusAndRetryCount Method 187
IAsptNotification 188
IAsptNotification::NotificationID Property 188
IAsptNotification::Status Property 188
IAsptNotification::SubmissionDate Property 189
IAsptNotification::LastProcessedDate Property 190
IAsptNotification::DispatcherID Property 190
IAsptNotification::RetryCount Property 190
IAsptNotification::RetryInterval Property 191
IAsptNotification::ContactInfoProvider Property 191
IAsptNotification::RecipientCount Property 192
IAsptNotification::DispatchType Property 192
IAsptNotification::Recipients Property 193
IAsptNotification::NotificationMessage Property 193
IAsptNotification::SenderID Property 193
IAsptNotification::AddRecipientID Method 194
IAsptNotification::RemoveRecipient Method 194
IAsptNotification::RemoveDuplicateRecipients Method 195
IAsptNotificationAttachment 195
IAsptNotificationAttachment::NotificationAttachmentID Property 195
IAsptNotificationAttachment::NotificationID Property 196
IAsptNotificationAttachment::FilePathname Property 196
IAsptNotificationAttachment::MediaType Property 197
IAsptNotificationAttachment::DispType Property 197
IAsptNotificationDispatcher 197
IAsptNotificationDispatcher::DispatcherType Property 198

IAsptNotificationDispatcher::WorkQueueSize Property 198
IAsptNotificationDispatcher::DispatchNotification Method 198
IAsptNotificationDispatcher::DispatchNotificationWithAttachment Method 199
IAsptNotificationXMLGenerator 200
IAsptNotificationXMLGenerator::GenerateXML Method 201
IAsptNotificationXMLGenerator::GenerateXMLWithAttachment Method 201
IAsptNotificationStreamProvider 202
IAsptNotificationStreamProvider::CLSID Property 202
IAsptNotificationStreamProvider::CreateStream Method 202
IAsptNotificationStream 203
IAsptNotificationStream::ReadProperty Property 203
IAsptNotificationStream::WriteNotification Method 204

Work Rule API Developer Guide 206
Related Software Files 206
File List 206
Prerequisites 206
Additional Requirements 207
Audience 207
Additional Documentation 207
Overview 208
Components of the Work Rule API 208
Installing the Work Rule API 208
Concepts in Aspect Workforce 208
Trial Schedules 208
Official Schedules 209
Work Rules 209
Work Rule Sets 209
Work Rule Plug-ins 209
Work Rule Evaluation of Schedules 210
Work Rule Internal State 211
Work Rule Life Cycle during Scheduling 212
Automatic Assignment of Trial Schedules 214
Preference-Based Scheduling 214
Fiscal Calendars 215
Data Types and Stored Procedures 215
Database Tables 215
RULE_PLUG_IN 215

FSC_CAL 216
FSC_CAL_DET 217
Work Rule Plug-in COM Interfaces Reference 217
ISegment Interface 217
ISegment Properties 218
IDetailSegment Interface 218
IDetailSegment Properties 218
IGeneralSegment Interface 219
IDaySchedule Interface 219
IDaySchedule Properties 220
IDaySchedule DetailSegment Method 221
IDaySchedule GeneralSegment Method 221
IDayScheduleSet Interface 221
IDayScheduleSet DaySchedule Method 223
IDayScheduleSet DayScheduleByNominalDate Method 223
IDayScheduleSetList Interface 223
IDaySCheduleSetList DayScheduleSet Method 224
IWorkRule Interface 224
IWorkRule GetCredentials Method 226
IWorkRule GetConfigurationInfo Method 226
IWorkRule InitializeWorkRule Method 228
IWorkRule FinalizeWorkRule Method 228
IWorkRule BeginScheduleScope Method 229
IWorkRule EndScheduleScope Method 229
IWorkRule BeginEmployeeScope Method 229
IWorkRule EndEmployeeScope Method 230
IWorkRule EvaluateDayScheduleSet Method 230
IWorkRule AddDayScheduleSet Method 230
IWorkRule RemoveDayScheduleSet Method 231
Work Rule Scheduling Scoring 231
Work Rule Scoring Example 231
Pass / Fail Work Rules 235
Work Rule Plug-in Implementation Steps 235
Performance Guidelines 236
Work Rule Plug-in Integration 236
Work Rule Plug-in Registration 237
Example Script for SQL Server 237

Example Script for Oracle 238
Work Rule Plug-in Installation 238
Work Rule Plug-in Example 239
SampleWorkRule.csproj 239
SampleWorkRule.cs 239
SampleWorkRuleConfig.cs 239
SampleWorkRuleCredentials.cs 239
SampleWorkRuleConfigEditor 239
SampleDayScheduleProperties.cs 240
SampleDayScheduleSetProperties.cs 240
SampleScheduleContext.cs 240
SampleEmployeeContext.cs 240
SampleEvaluationContext.cs 240
SampleScoreCalculator.cs 240

Using the Web Services 241
Prerequisites 241
Launching the Online Help 241
About the Web Services Servers 242
Using the Web Services for Specific Tasks 243
How to Make a Web Service Request 243
Installing the Web Services 243
Sample Script 243
Customizing the Sample Script 246
Recommended XML Coding Practices 247

Controller Tester User Guide 249
Prerequisites 249
Audience 249
Overview 249
Launching the Utility 250
About the Main Window 250
Service Selection Panel 250
Tabs 251
Session Tab 251
Data In Tab 251
Data Out Tab 251
Notes Tab 251

Menu Bar and Toolbar 252
File Menu Options 253
XML Menu Options 253
Edit Menu Options 254
View Menu Option 254
Tools Menu Options 254
Help Menu Option 259
Status Bar 259
Choosing Servlet Names with HTTP 259
Intra-Day Performance Column Descriptions 261
Audience 261
Overview 261
About the IDP Web Service 261
Using the IDP Web Service with Encompass 261
Information Provided 262
Forecast Group Statistics 262
AABD Actual Calculated Abandoned 262
AAHT Actual AHT 262
ABCKLOG Actual Calculated Backlog 263
ABCKLOG J Actual Calculated Backlog with Staff Adjustments 263
ABD-ACD Actual Abandoned – From ACD 264
Actual Simulated Queue Backlog With Adjustments 264
Actual Simulated Queue Backlog Without Adjustments 265
ADELAY HR Actual Calculated Average Delay 2: Hours 265
ADELAY J HR Actual Calculated Average Delay 1 (with Staff Adjustments): Hours 266
ADELAY J MIN Actual Calculated Average Delay 1 (with Staff Adjustments): Minutes 266
ADELAY J SEC Actual Calculated Average Delay 1 (with Staff Adjustments): Seconds 267
ADELAY MIN Actual Calculated Average Delay 2: Minutes 268
ADELAY SEC Actual Calculated Average Delay 2: Seconds 269
ANCA Actual Outbound Attempts 269
ANCH Actual Number of Contacts Handled 270
ANCO Actual Number of Contacts Offered 270
ANNC Actual Non-Connections 271
AOABDPCT Actual Outbound Abandoned Percent 271
ARPC Actual Right-Party Contacts 272
ARPCATTPCT Right-Party Contacts to Attempts Percent 273
ARPCCONPCT Right-Party Contacts to Total Connections Percent 273

ARPHT Actual Right-Party Average Handle Time 274
ASL Actual Calculated Service Level Percent 2 274
ASL J Actual Calculated Service Level Percent 1 (with Staff Adjustments) 275
ASLSEC Actual Service Level Seconds 276
ATPC Actual Total Contacts 276
AWPC Actual Wrong-Party Contacts 277
AWPCATTPCT Wrong-Party Contacts to Attempts Percent 277
AWPCCONPCT Wrong-Party Contacts to Connections Percent 278
AWPHT Actual Wrong-Party Average Handle Time 279
BACKLOG Actual Email Backlog 279
Balk Percentage 280
DELAY-ACDHR Actual Average Delay: Hours 280
DELAY-ACDMIN Actual Average Delay: Minutes 281
DELAY-ACDSEC Actual Average Delay: Seconds 281
DELAYGOALHR Average Delay Goal: Hours 282
DELAYGOALMIN Average Delay Goal: Minutes 282
DELAYGOALSEC Average Delay Goal: Seconds 283
EXCL Exclude from Reforecasting Indicator 283
FGANET Forecast Group's Actual Net Staff 2 284
FGANET J Forecast Group's Actual Net Staff 1 (with Staff Adjustments) 285
FGANETC Forecast Group’s Actual Net Calls 285
FGANETC J Forecast Group’s Actual Net Calls (with Staff Adjustments) 286
FGAREQ Forecast Group's Actual Required without Unproductive 287
FGAREQC Forecast Group’s Actual Required Calls 287
FGAREQC J Forecast Group’s Actual Required Calls with Staff Adjustments and without Unproductive 288
FGASCH Forecast Group's Actual Scheduled without Staff Adjustments and Unproductive 289
FGASCH J Forecast Group's Actual Scheduled with Staff Adjustments and without Unproductive 289
FGLOCK Forecast Group's ACD Data Lock Indicator 290
FGNOTE Forecast Group Note 291
FGSTATUS Forecast Group's ACD Data Status Indicator 291
FGONET Forecast Group's Original Net Staff 292
FGONETC Forecast Group’s Original Net Calls 292
FGOREQ Forecast Group's Original Required without Unproductive 293
FGOREQC Forecast Groups’s Original Required Calls 294
FGOSCH Forecast Group's Original Scheduled without Unproductive 294
FGRNET Forecast Group's Revised Net Staff 2 295

FGRNET J Forecast Group's Revised Net Staff 1 (with Staff Adjustments) 296
FGRNETC Forecast Group's Revised Net Calls 297
FGRNETC J Forecast Group's Revised Net Calls (with Staff Adjustments) 297
FGRREQ Forecast Group's Revised Required without Unproductive 298
FGRREQC Forecast Group's Revised Required Calls 299
FGRREQC J Forecast Group's Revised Required Calls (With Staff Adjustments) 300
FGRSCH Forecast Group's Revised Scheduled without Staff Adjustments and Unproductive 300
FGRSCH J Forecast Group's Revised Scheduled with Staff Adjustments and without Unproductive 301
OAAHTDEV Original vs. Actual AHT Deviation 302
OAAHTPDEV Original vs. Actual AHT Percent Deviation 302
OAAHTRTIO Original vs. Actual AHT Ratio 303
OABD Original Forecast Calculated Abandoned 303
OAHT Original Forecast AHT 304
OAVOLDEV Original vs. Actual Volume Deviation 305
OAVOLPDEV Original vs. Actual Volume Percent Deviation 305
OAVOLRTIO Original vs. Actual Volume Ratio 306
OBCKLOG Original Forecast Calculated Backlog 306
ODELAYHR Original Calculated Average Delay: Hours 306
ODELAYMIN Original Calculated Average Delay: Minutes 307
ODELAYSEC Original Calculated Average Delay: Seconds 308
Original Simulated Queue Backlog 308
OSL Original Calculated Service Level Percent 309
OVOL Original Forecast Volume 309
RAAHTDEV Revised vs. Actual AHT Deviation 310
RAAHTPDEV Revised vs. Actual AHT Percent Deviation 310
RAAHTRTIO Revised vs. Actual AHT Ratio 311
RABD Revised Forecast Calculated Abandoned without Staff Adjustments and Unproductive 312
RABD J Revised Forecast Calculated Abandoned with Staff Adjustments and without Unproductive 312
RAHT Revised Forecast AHT 313
RAHTTS Revised Forecast AHT Update Timestamp 313
RAVOLDEV Revised vs. Actual Volume Deviation 314
RAVOLPDEV Revised vs. Actual Volume Percent Deviation 314
RAVOLRTIO Revised vs. Actual Volume Ratio 315
RBCKLOG Revised Forecast Calculated Backlog 316

RBCKLOG J Revised Forecast Calculated Backlog with Staff Adjustments 316
RDELAY HR Revised Calculated Average Delay 2: Hours 316
RDELAY J HR Revised Calculated Average Delay 1 (with Staff Adjustments): Hours 317
RDELAY J MIN Revised Calculated Average Delay 1 (with Staff Adjustments): Minutes 317
RDELAY J SEC Revised Calculated Average Delay 1 (with Staff Adjustments): Seconds 318
RDELAY MIN Revised Calculated Average Delay 2: Minutes 319
RDELAY SEC Revised Calculated Average Delay 2: Seconds 319
Revised Forecast Handling Capability Without Adjustments 320
Revised Simulated Queue Backlog With Adjustments 320
Revised Simulated Queue Backlog Without Adjustments 321
Revised Volume Handling Capability With Adjustments 321
RSL Revised Calculated Service Level Percent 2 322
RSL J Revised Calculated Service Level Percent 1 (with Staff Adjustments) 322
RVOL Revised Forecast Volume 323
RVOLTS Revised Forecast Volume Update Timestamp 324
Simulated Actual Scheduled With Adjustments Without Unproductive 324
Simulated Original Scheduled Without Unproductive 325
Simulated Revised Scheduled With Adjustments Without Unproductive 325
Simulated Revised Scheduled Without Adjustments Without Unproductive 325
SL-ACD Actual Service Level 326
SLGOAL Service Level Goal: Percent 326
SLGOALHR Service Level Goal: Hours 327
SLGOALMIN Service Level Goal: Minutes 328
SLGOALSEC Service Level Goal: Seconds 328
TENDENCYTOABANDON Tendency to Abandon 329
WRAP Wrap-up Seconds 330
Staff Group Statistics 330
AABSREQADH 1 Absolute Actual Required Adherence 1 330
AABSREQADH 2 Absolute Actual Required Adherence 2 331
ABSSCHADH 1 Absolute Scheduled Adherence 1 332
ABSSCHADH 2 Absolute Scheduled Adherence 2 332
ABSSCHADH 3 Absolute Scheduled Adherence 3 333
AGGFGRSCH Forecast Group's Revised Aggregated Scheduled without Staff Adjustments and Unproductive 334
APS Actual Positions Staffed 334
AREQADH1 Actual Required Adherence 1 335
AREQADH2 Actual Required Adherence 2 336

AREQDEV 1 Actual Required Deviation 1 336
AREQDEV 2 Actual Required Deviation 2 337
MAXOCC Maximum Agent Occupancy 338
OABSREQADH Absolute Original Required Adherence 338
OREQADH Original Required Adherence 339
OREQDEV Original Required Deviation 339
Other Shrinkage Percent 340
RABSREQADH 1 Absolute Revised Required Adherence 1 340
RABSREQADH 2 Absolute Revised Required Adherence 2 341
RREQADH 1 Revised Required Adherence 1 342
RREQADH 2 Revised Required Adherence 2 343
RREQDEV 1 Revised Required Deviation 1 343
RREQDEV 2 Revised Required Deviation 2 344
RREQTS Revised Required Update Timestamp 345
SCHADH 1 Scheduled Adherence 1 345
SCHADH 2 Scheduled Adherence 2 346
SCHADH 3 Scheduled Adherence 3 346
SCHDELTA Schedule Tally Delta 347
SCHDEV 1 Scheduled Deviation 1 347
SCHDEV 2 Scheduled Deviation 2 348
SCHDEV 3 Scheduled Deviation 3 349
SGANET Staff Group's Actual Net Staff 2 349
SGANET J Staff Group's Actual Net Staff 1 350
SGAREQ Staff Group's Actual Required without Staff Adjustments and Unproductive 351
SGAREQ J Staff Group's Actual Required with Staff Adjustments and without Unproductive 351
SGAREQ JU Staff Group's Actual Required with Staff Adjustments and Unproductive 352
SGAREQ U Staff Group's Actual Required without Staff Adjustments and with Unproductive 353
SGLOCK Staff Group's ACD Data Lock Indicator 353
SGNOTE Staff Group Note 354
SGONET Staff Group's Original Net Staff 354
SGOREQ Staff Group's Original Required without Staff Adjustments and Unproductive 355
SGOREQ U Staff Group's Original Required without Staff Adjustments and with Unproductive 356
SGOSCH Staff Group's Original Scheduled 356
SGRNET Staff Group's Revised Net Staff 2 357
SGRNET J Staff Group's Revised Net Staff 1 358

SGRREQ Staff Group's Revised Required without Staff Adjustments and Unproductive 359
SGRREQ J Staff Group’s Revised Required with Staff Adjustments and without Unproductive 359
SGRREQ JU Staff Group's Revised Required with Staff Adjustments and Unproductive 360
SGRREQ U Staff Group's Revised Required without Staff Adjustments and with Unproductive 361
SGRSCH Staff Group's Revised Scheduled without Staff Adjustments 361
SGRSCH J Staff Group's Revised Scheduled with Staff Adjustments 362
SGSTATUS Staff Group's ACD Data Status Indicator 363
Stored Original Required Without Adjustments Without Unproductive 363
Stored Revised Required Without Adjustments Without Unproductive 364
UNP Unproductive Shrinkage Percentage 364

Appendix A. Security Auditing 366
Security Tables Audited 366
Audit Trail Tables 366
Security Violation Tables 367

Appendix B. Glossary of Aspect Workforce Terms 369

Revision History

Date	Description	Section
8/252023	Rev A, initial Release	NA
10/10/2023	Rev B	Table column heading corrections, pgs 181-339
10/4/2024	Rev C	Aspect rebranding
11/7/2025	Rev D	Updated Legal Notices and back page matter for 2025 Update filepaths to reflect 25.0 Update Windows Server versions to 2022 and 2025.

About this Guide

This guide is part of the Aspect Workforce™ – Software Development Kit (SDK). The SDK provides documentation, reference materials, programming examples, and a utility that demonstrate the use of the components of Aspect Workforce™.

Audience

The guide is intended for application developers who want to extend the capabilities of Aspect Workforce by creating custom components with the SDK. The following skills are required of the user:

- Familiarity with Aspect Workforce
- Entry-level programming knowledge

Programming examples are given in several formats, including Visual C++, HTML, and XML.

Organization of this Guide

This guide consists of the following sections: **API Guides**, documents the APIs for creating Aspect Workforce applications.

1. Tally Server API Developer Guide, introduces the Tally Server API and describes the use of example programs provided with this SDK. Applications created with this API enable you to query the Aspect Workforce system for information related to agents' schedules.

2. Updater API Developer Guide, specifies the Updater API for development of third-party applications with Aspect Workforce. The Updater API enables you to add, make changes to, and delete segments that comprise agents' schedules.

3. Updater Plug-In Rule Developer Guide, specifies the information necessary to create, implement, and install an Updater plug-in rule in Aspect Workforce. Updater plug-in rules enhance the native functionality of the Updater system service of Aspect Workforce.

4. Notification Server API Developer Guide, describes the APIs provided by Notification Server for use in developing custom applications in terms of properties and methods. This API assists with the automatic issuance of staffing-related change notifications to affected parties.

5. Work Rule API Developer Guide describes the APIs for development of third-party work rule plug-ins. Work rules constrain schedule assignment during automatic trial schedule assignment and preference-based scheduling to prevent rule violations.

6. Using the Web Services, documents Aspect Workforce web services and provides related information, including how to make a web service request.

7. Controller Tester User Guide, describes the Aspect Workforce Controller Tester, a utility that enables developers to interact with the Aspect Workforce web services.

8. Intra-Day Performance Column Descriptions, This reference guide provides detailed descriptions of intra-day performance (IDP) columns that appear in Aspect Workforce. The column descriptions are needed to make requests when using the web services associated with the IDP server.

9. Appendix A. Security Auditing, provides details about the security tables audited, audit trail tables, and security violation tables.

10. Appendix B. Glossary of Aspect Workforce Terms, defines Aspect Workforce terms that are relevant to this SDK.

About Backward Compatibility

The SDK maintains backward compatibility with previous versions of the SDK, with the following exceptions:

- Two rarely used web services, IDPMStaffAdjustmentLookup and IDPMStaffAdjustmentUpdate, were deprecated in Aspect Workforce 7.4. If you have created custom applications that use these web services, the applications will no longer work correctly in Aspect Workforce 7.4 and later versions. Please contact Alvaria Customer Care for assistance.
- In Aspect Workforce 8.2 and later, the conversion of some of the applications and services to 64-bit required that the packing for many of the COM interfaces be changed from 4 bytes to 8 bytes. While we have tried to minimize the effect of this, some applications may need to be changed.
- In Aspect Workforce 22.0 and later, several changes were made to the Tally Server COM interfaces. The majority of these changes were related to new options for superstate filtering. Due to these changes, any applications that use a Tally Server COM interface that were developed prior to version 22.0 will need to be updated to be compatible with version 22.0 and higher.
- The Aspect Workforce public data model is compatible with one and only one version of Aspect Workforce. For example, version 22.0 of the public data model is compatible with Aspect Workforce 22.0. As new versions of Aspect Workforce are released, accompanying versions of the public data model are released with them. Typically, the differences between public data model versions are minor and would not affect custom applications that access the database directly. But in some cases, the differences will require modifications to custom applications.



Note: In general, if the desired functionality is provided by web services, then using the web services should be favored over directly accessing the database. Use of the web services will minimize anomalies relating to backward compatibility.

- Schema changes may require minor updates to custom applications. Please refer to the XML schema definitions in the web services Help (included with this SDK) for details about the correct format of XML inputs and outputs. Then update your custom applications accordingly. For more information, see Chapter 6, Using the Web Services.

Installing the SDK

The product CD includes a wizard that installs the entire SDK, including software and documentation.

Install the SDK by inserting the product CD and launching the installation program, **Setup.exe**. A wizard then guides you through the installation.

API Guides

This section documents the APIs for creating Aspect Workforce applications.

Tally Server API Developer Guide

This guide introduces the Tally Server API and describes the use of example programs provided with this SDK. The Tally Server API enables you to query the Aspect Workforce system for information related to agents' schedules.

Related Software Files

The SDK includes example programs and related files that demonstrate the functionality of the Tally Server API. After you install the SDK, these files can be found at the following default path:

```
C:\Program Files (x86)\Alvaria\Workforce 25.0 SDK\API_Related_Files\
Tally_Server_API_Related_Files
```

File List (Partial)

Related software files delivered with the SDK include the following:

- CLAS_DOWFilter.h
- CLAS_DOWFilter_fwd.h
- CLAS_DurationFilter.cpp
- CLAS_DurationFilter.h
- CLAS_DurationFilter_fwd.h
- CLAS_KeyLists.h
- CLAS_Moments.cpp
- CLAS_Moments.h
- CLAS_Moments_fwd.h
- CLAS_NomDate.cpp
- CLAS_NomDate.h
- CLAS_NomDate_fwd.h
- CLAS_ScheduleResolutionDefinition.cpp
- CLAS_ScheduleResolutionDefinition.h
- CLAS_ScheduleResolutionDefinition_fwd.h
- CLAS_SegmentSelector.cpp

- CLAS_SegmentSelector.h
- CLAS_SegmentSelector_fwd.h
- CLAS_SuperStateWithID.cpp
- CLAS_SuperStateWithID.h
- CLAS_SuperStateWithID_fwd.h
- CLAS_TallyDefinition.cpp
- CLAS_TallyDefinition.h
- CLAS_TallyDefinition_fwd.h
- CLAS_TallyExceptions.h
- CLAS_TallyTypeReleases.cpp
- CLAS_TallyTypeReleases.h
- CLAS_TCSConstants.h
- CLAS_TimeFilter.cpp
- CLAS_TimeFilter.h
- CLAS_TimeFilter_fwd.h
- StringFunctions.cpp
- StringFunctions.h
- TallyServerSampleCPP.cpp
- TallyServerSampleCPP.vcxproj
- TallyServerSampleCPP.vcxproj.filters
- TallyTransport.h
- TallyTransport.idl
- TallyTransport_i.c
- TallyTransport_p.c
- TallyTypes.h
- TallyTypes.idl
- TCSJobCommon.h
- TCSJobCommon.idl

Prerequisites

The following table shows the requirements for using the Tally Server API.

Aspect Workforce Requirements for the Tally Server API

Versions	Any version of Tally Server 7.5 or later. The Tally Server interfaces in version 22.0 are no longer compatible with prior versions. If you developed an application that uses any of the Tally Server interfaces from a version

	prior to 22.0, you will need to update that application to be compatible with 22.0. The IDL files delivered with the SDK describe the updated interfaces.
Enhancement Packages	None

Additional Requirements

In addition to the prerequisites, the Tally Server API has the following requirements relating to hardware, software, connectivity, and security:

- A computer that meets the requirements for the Aspect Workforce client software.
- The Aspect Workforce client installed on the computer from where you will use the Tally Server API.
- While the Tally Server API does not run the Aspect Workforce client (TCS.EXE), it does require several files included in the Aspect Workforce installation.
- Network connections to the Aspect Workforce database server and application server(s).
- An Aspect Workforce username and password for use with the Tally Server API. If you do not have an Aspect Workforce account, you must ask your administrator to create one for you

Audience

This guide assumes that you are familiar with the following:

- Basic database concepts
- SQL programming
- COM concepts and programming methods
- Visual C++

For more information on the above topics please consult the documentation provided with the respective products or the large number of books available on the topics.

Additional Documentation

The documentation for the Tally Server API consists of this developer guide. However, general Aspect Workforce concepts are explained in the Aspect Workforce User Guide, and the Aspect Workforce online Help.

Overview

This reference guide specifies the application programming interface (API) that third-party developers can use to query the Aspect Workforce system for information related to agents' schedules. The Tally Server API is used by the Aspect Workforce client to provide statistics and state information related to agents' schedules. The same API, documented here, can be used by application developers at a customer site to extract schedule information while still utilizing the segment priority rules enforced by the Tally Server. The Tally Server is one of several services that comprise the Aspect Workforce application server. Aspect Workforce is a comprehensive resource planning and workforce management system that gives you the tools you need to achieve your sales or service goals at the lowest cost. Designed especially for customer contact centers, it also has applications in other kinds of work centers—such as commercial airports—

where there are continuous, but varying, streams of work that must be promptly handled by large numbers of staff.

Tally Server performs processor-intensive complex business logic. The most complex of Tally Server's computations is the tally of groups of schedules by period of day. This requires analyzing the schedules of all employees in the selected group for the selected time period, including their work, break, and off-phone segments such as meetings, projects, and vacations. Tally Server resolves the overlaps among segments and determines the contribution of each schedule to a staff group for each time period. The processing can be extensive if there are many agents, each having many segments in each schedule. The solution is to set up the Tally Server service to maintain a continuous tally for selected Staff Groups and a selected relative number of days. Tally Server maintains an updated tally by receiving notification from Updater whenever there are changes to an official schedule. Tally Server also resolves individual schedules for display by the client and resolves segments for summarizing different types of schedule hours. The Aspect Workforce client and some other services request and receive data from Tally Server, while Tally Server talks to the database server directly.

Tally Server Communications

Tally Server may be set up to run on multiple application server machines. Each machine would be configured for Tally Server to maintain a continuous tally of specified staff groups for specified relative dates (usually, yesterday, today, and tomorrow). This means that Tally Server holds tally data up to date in memory for clients to experience faster response time. Client machines are set up to communicate with a single Tally Server machine, according to the primary staff group tallies that the user accesses most frequently. Note, however, that the Tally Server service can provide a tally for a staff group whether or not it is holding that staff group's tally up to date in memory. This means that a client can request a tally for any staff group from the designated Tally Server machine.

Tally Server communicates with the following entities:

- Aspect Workforce client
- Real-Time Adherence service
- Administrators

Communication between Tally Server and each entity is described below.

Aspect Workforce Client

The following components of the Aspect Workforce client communicate with Tally Server:

- Intra-day Time Line (requests schedule resolution data)
- Intra-day Performance (IDP) (requests schedule tally data by staff group)
- Allocation Calculator (creates an IDP tally object, which requests schedule tally data)
- Meeting Optimization (requests schedule tally from Tally Server)
- Break Optimization (requests schedule tally from Tally Server)
- Project Optimization (requests schedule tally from Tally Server)
- Scheduler (requests a tally of Base Staff when creating schedules or running schedule tests)
- Agent Productivity (requests schedule resolution data when the user creates an Agent Availability report)

Real-Time Adherence Service

The Real-Time Adherence service communicates with Tally Server by requesting schedule resolution data for all employees once per day, on startup, and as requested by Real-Time Adherence.

Administrators

Administrators communicate with Tally Server by requesting schedule resolution data for each employee whose schedule data changes when Real-Time Adherence receives notification from Updater.

API Goals and Assumptions

The Tally Server API is designed to provide an environment in which users can obtain state information for an employee's schedules and related statistics. The Tally Server server ensures that the various rules regarding segment priorities defined by the Aspect Workforce administrator are always enforced. The Tally Server API also allows for the integration of third-party developed applications with the Aspect Workforce system. These might include applications that require official schedule information that is needed by time-keeping systems, automated attendance tracking systems, etc. Before using the Tally Server API you must be familiar with the basic concepts of schedules and segments in Aspect Workforce. This document outlines some of concepts that are fundamental in understanding schedules and the way that they are stored and manipulated in Aspect Workforce.

Tally Server is a DCOM service running on one or more of the Aspect Workforce application servers. The Aspect Workforce client connects to the default Tally Server as defined in the registry.

Tally Server Request Types

Tally Server handles the following five types of requests:

- A moment-based schedule resolution
- A moment-based time period tally
- A nominal-date-based schedule list tally
- A nominal-date-based schedule tally
- A moment-based schedule list tally

Each type of request is described in the following sections.

Moment-Based Schedule Resolution

The moment-based schedule resolution queries the super-schedule resolution data by the moment ranges specified in the request.

Sample results for moment-based schedule resolution

Employee: 1952

Perspective: STAFF

Default State: FREE

Moment-Based Schedule Resolution (Sample Results)

Time Period	Start Moment	Stop Moment	Nominal Date	State
0	1/9/2001 2:45 AM	1/9/2001 4:45 AM	1/9/2001	TCSTCS

Time Period	Start Moment	Stop Moment	Nominal Date	State
1	1/9/2001 4:45 AM	1/9/2001 5:00 AM	1/9/2001	BREAK
2	1/9/2001 5:00 AM	1/9/2001 7:45 AM	1/9/2001	TCSTCS
3	1/9/2001 7:45 AM	1/9/2001 8:45 AM	1/9/2001	BREAK
4	1/9/2001 8:45 AM	1/9/2001 10:00 AM	1/9/2001	TCSTCS

Moment-Based Time Period Tally

The moment based time period tally queries the super-schedule superstate resolution by the moment ranges in the request to determine the number of schedules that evaluate to true at any given moment for each superstate in the request.

Number of Tallies: 1

Results for SuperStateID 0

Moment-Based Time Period Tally (Sample Results)

Time Period	Start Moment	Stop Moment	Head Count	Total Minutes
0	1/7/2001 3:30 PM	1/7/2001 5:15 PM	1	105
1	1/7/2001 5:30 PM	1/7/2001 7:30 PM	1	120
2	1/7/2001 7:30 PM	1/7/2001 8:00 PM	2	60
3	1/7/2001 8:00 PM	1/7/2001 8:15 PM	1	15
4	1/7/2001 8:15 PM	1/7/2001 10:00 PM	2	210

Nominal-Date-Based Schedule List Tally

The nominal-date-based schedule list tally is a direct representation of the super-schedule superstate resolution.

Sample results for Schedule 2

Employee: 1952

Results for SuperStateID 0

Nominal-Date-Based Schedule List Tally (Sample Results)

Time Period	Start Moment	Stop Moment	Nominal Date
0	1/9/2001 2:45 AM	1/9/2001 4:45 AM	1/9/2001
1	1/9/2001 5:00 AM	1/9/2001 7:45 AM	1/9/2001
2	1/9/2001 8:45 AM	1/9/2001 10:00 AM	1/9/2001
3	1/9/2001 10:15 AM	1/9/2001 11:45 AM	1/9/2001
4	1/10/2001 2:45 AM	1/10/2001 4:45 AM	1/10/2001

Nominal Date Based Schedule Tally

The nominal date based schedule tally queries the super-schedule superstate resolution by nominal date ranges to determine the number of minutes in each nominal date that evaluate to true for each superstate in the request.

Sample results for Schedule 2

Employee: 1952

Results for SuperStateID 0

Nominal Date-Based Schedule Tally (Sample Results)

Time Period	Nominal Date	Total Minutes
0	1/9/2001	450
1	1/10/2001	450
2	1/11/2001	450
3	1/12/2001	450
4	1/13/2001	450

Moment-Based Schedule List Tally

The moment-based schedule list tally is a direct representation of the super-schedule superstate resolution.

Sample results for Schedule 2

Employee: 1952

Results for SuperStateID 0

Moment-Based Schedule List Tally (Sample Results)

Time Period	Start Moment	Stop Moment	Nominal Date
0	1/9/2001 2:45 AM	1/9/2001 4:45 AM	1/9/2001
1	1/9/2001 5:00 AM	1/9/2001 7:45 AM	1/9/2001
2	1/9/2001 8:45 AM	1/9/2001 10:00 AM	1/9/2001
3	1/9/2001 10:15 AM	1/9/2001 11:45 AM	1/9/2001
4	1/10/2001 2:45 AM	1/10/2001 4:45 AM	1/10/2001

Installing the API

Installing the Tally Server API is a straightforward process. Perform the following steps on the computer where you want to access the Tally Server API:

1. Install the Aspect Workforce client software. This is done by running the setup.exe program from the WFMSetup/Workforce Management share on your application server.

2. Create a project in Visual C++. An example of code is provided later in this chapter. The sample C++ project provided with this SDK includes a number of files that you may find useful.

3. Install the ODBC driver for your Aspect Workforce database.

4. Create an ODBC User or System DSN to connect to your Aspect Workforce database. This will be needed to select from Aspect Workforce tables for surrogate key information in order to populate request structures.

Concepts in Aspect Workforce

There are several fundamental Aspect Workforce concepts that you must understand before you can successfully use the Tally Server API. Some of these concepts are also explained in the online Help.

Surrogate Keys (SK)

The primary key in most of the Aspect Workforce tables is a surrogate key (SK). Surrogate keys are defined as the NUMERIC data type and must be obtained from either the GET_SK or the GET_SKS stored procedures. The extensive use of surrogate keys in the Aspect Workforce data model permits the easy changing of properties that Aspect Workforce users normally identify as unique, which are stored instead as alternate keys in the database. For example, the primary key in the employee table (EMP) is a generated surrogate key whereas the employee id field (ID), supplied by the user, is an alternate key. It is the primary key of the employee table (EMP_SK), which appears in composite alternate and composite foreign keys in other tables. Hence, the user can safely modify the employee's ID without jeopardizing the integrity of the data in any of the other tables. Surrogate keys are not revealed to the user in the Aspect Workforce client software and are of use only to someone who needs to access or modify Aspect Workforce data directly or through the Tally Server API.

In the database, they are represented as a integer number. Because of the allowed range of these integers, we use double precision (64-bit) floating point numbers to hold these values. A double precision floating-point number allows for 48-bit integer values that are sufficient to hold the numbers in the database.

Within Aspect Workforce code, when an invalid key value is needed, the value 0.5 is used. This constant is also known as BadKey. This value should not ever be seen in the database.

Perspectives, States, Segment Codes, Superstates

These entities are basic to Aspect Workforce and are described below. Additionally, the Aspect Workforce User Guide contains more detail on setting up these items in Aspect Workforce.

Perspectives

Perspectives track the multiple implications of a single segment in terms of:

- ACD login status
- Pay status
- Staffing status
- Coaching status
- Other user-defined perspectives

The following table shows the four pre-populated perspectives, a description of each, and the various states associated with those perspectives. The default state for that perspective is in italics.

Prepopulated Perspectives With Their Associated States

Code	Description	States (default is in italics)
STAFF	Work status	*FREE*, BREAK, IDLE, LUNCH, VACATN, MEETNG, TRAING, PROJCT, SICK, LEAVE, TYPE OF WORK
PAY	Pay status	*UNPAID*, NORMAL, OVERTM
ACD	ACD login status	*NO*, YES
COACH	Working with help	*NO*, YES

Perspective of Basic Shift Segment

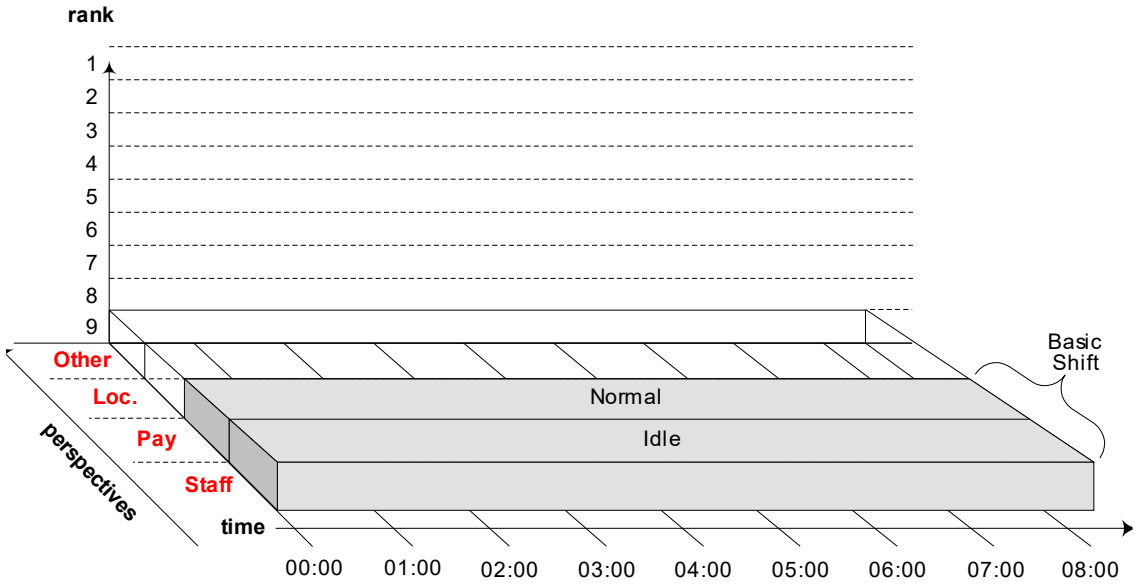
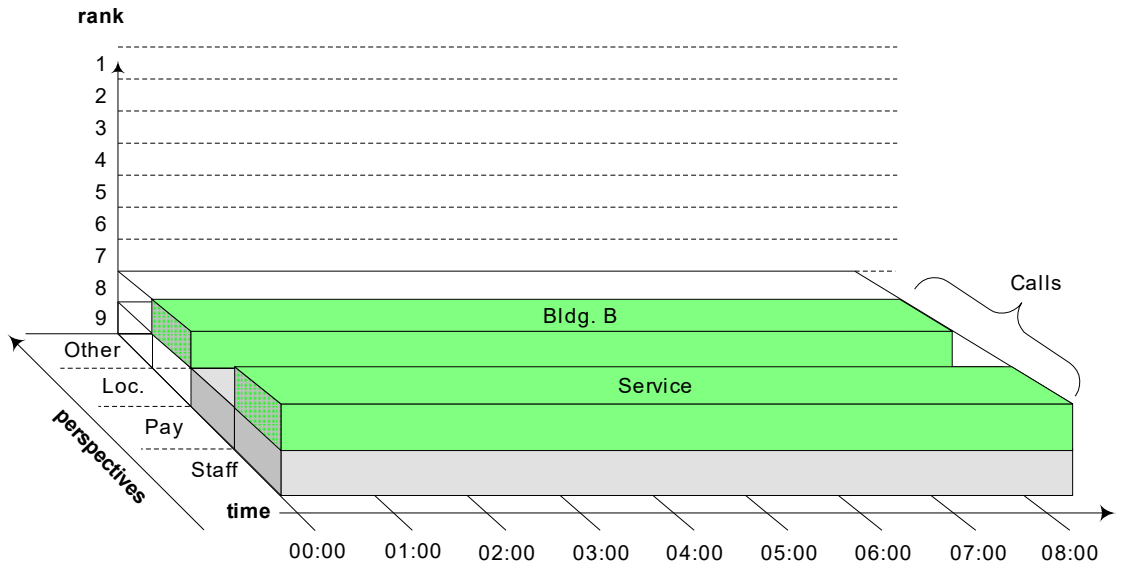


Figure-1 Shows the basic shift segment as seen from the perspectives of Staff, Pay, Location, and Other.

Figure-1: Perspectives of Basic Shift Segment

Perspectives of Call Handling Segment

As shown in Figure Figure-2, The call-handling segment has been laid over the basic shift segment. We see the call-handling segment from the perspectives of Staff, Pay, Location, and Other.



Figure

Figure-2: Perspectives of Call Handling Segment

Perspectives and Ranking

As shown in Figure-3, Two break segments, a lunch segment, and a training segment have been added. The segments are shown in the Staff, Pay, Location, and Other perspectives. If a segment is shaded in a given perspective, it takes priority over the segments below it. If a segment is unshaded in a given perspective, it does not take priority over the segments below it.

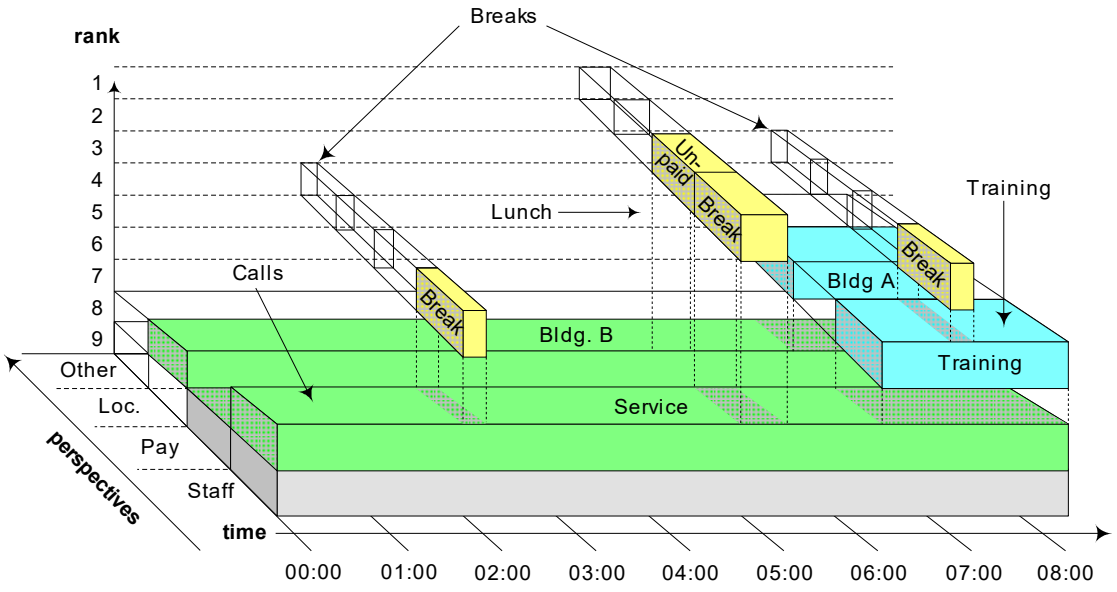


Figure-3: Perspectives and Ranking

Perspectives from the Top-Down View

As shown in Figure-4, each perspective has its own track on the timeline. The track shows the various states for each perspective throughout the day.

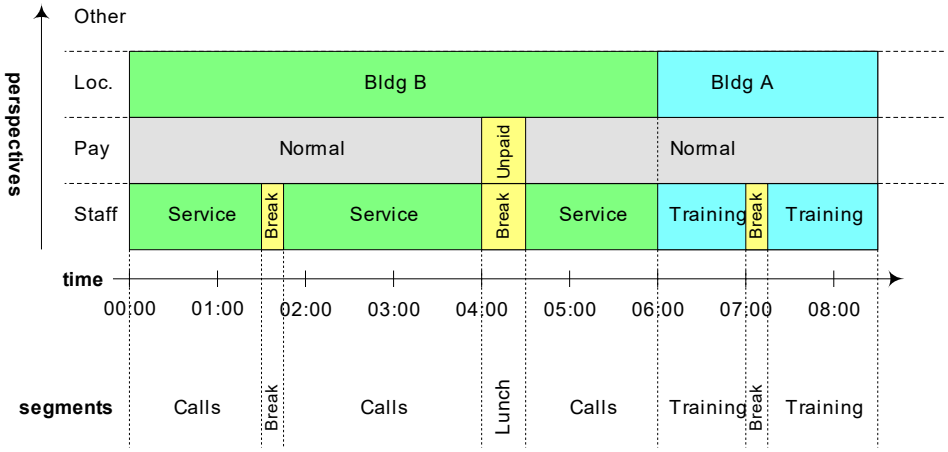


Figure-4: Perspectives from the Top-Down View

States

A state refers to the value, or effect, that a segment has with respect to a particular perspective.

For example, if an employee has a doctor's appointment at 10:00 a.m. and a DOCT segment is added, the states for that DOCT segment may be as shown in the following table.

Table -1: Perspectives with Sample States

Perspective	State
STAFF	SICK
PAY	UNPAID
ACD	NO
COACH	NO

Schedules and Segments

A schedule is a record of what an employee is supposed to do on any given day and is simply a collection of segments for that date. Segments are the basic building blocks of schedules. They are the essential records you create to track how employees are spending time and how well you are achieving your staffing objectives. To record the fact that an employee is attending a meeting, for example, you would enter a segment for that employee that includes the date, time period, and the appropriate segment code (defined later).

Segment Codes

Segment Codes are used to identify the type of activity that a segment represents. For example the segment code "UNPBRK" may be used to represent unpaid breaks. Each segment must be associated with a segment code.

Segment Categories

Segment categories are broad characterizations of segment codes that enable you to define generalized segment entry rules. They allow the definition of rules at a higher level than possible with just segment codes. For example, to set up a rule that forces segments of category “break” to be contained in segments of the category “work” you can associate all the break segment codes with the break category and all the work segment codes with the work category and them set up just one rule.

Segment Entry Rules

Segment entry rules are rules that you define which Tally Server enforces whenever a change or an addition is made to an employee’s official segment data. There are two categories of segment entry rules: containment rules and overlap rules. Segment containment rules allow you to define rules that force segments of a particular category to be completely contained in segments of another category. Segment overlap rules allow you to define rules that do not allow segments of a particular category to overlap segments of another category.

For example, you could define a containment segment entry rule that forced all segments that are of the category “TRNG” to be completely contained in segments that are of the category “WORK”. This would define a rule that would force all training related segments to be contained in work segments. Similarly, you could define an overlap segment entry rule that prohibited segments of category “TRNG” to overlap segment of category “VACA”, and so on.

Segment Types

There are two types of segments: General segments and Detail segments. Any segment code can be used to enter either a General or a Detail segment.

General Segments

A general segment is a segment that does not have a start or a stop moment but is associated with a particular nominal date. Any segment can be entered as a general segment, which is in effect whenever the employee has at least one detail segment for the same nominal date. General segments are used to denote “activity” that is valid for the entire day’s schedule for a person. For example, to denote that a person is on vacation for the day regardless any other activity previously assigned to the person, you could enter a General segment with a vacation code.

Detail Segments

A detail segment is a segment that is not only associated with a nominal date but also has a start and a stop moment. The start and stop moments delineate the absolute time that the segment is in effect.

Schedule Segments

Schedule segments are used to describe what activity an employee is scheduled to be doing. They come in two types: detail and general. Both types of segments have a unique key that identifies the segment, a key that points to a segment code or type and a nominal date. Detail segments also contain an absolute moment range. Detail segments are the type usually used to describe a schedule. General segments overlay all detail segments with the same nominal date. If there are no detail segments with the same nominal date as a general segment, the general segment will have no effect on a schedule.

Official Schedules

Official schedules are those schedules that are officially assigned to an employee. These schedules are the official historical record of what has happened in the past and the official statement about what is expected to happen in the future.

Trial Schedules

Trial schedules are the what-if schedules. They typically contain results of scheduling runs that may or may not be made official. Trial schedules are grouped into trial schedule sets.

Superstates

Superstates provide Aspect Workforce with a method of tallying employee time for tracking reports and other applications.

Aspect Workforce lets the user control the scope of the Superstates tally in the following ways:

- Broaden the scope to include time spent in several states from the same perspective. Example: All time spent in work states SICK and LEAVE.
- Narrow the scope to include only time spent in an overlap of states in different perspectives. Example: All time spent in pay state NORMAL and in work state VACATION.

All superstates are active. A system service running on the Aspect Workforce application server continually updates the tallies that are based on it.

In order for Updater to keep a superstate up to date, the “Accumulate summary hours data” check box must be checked.

Superstate: Example

Superstate = All PAID time off

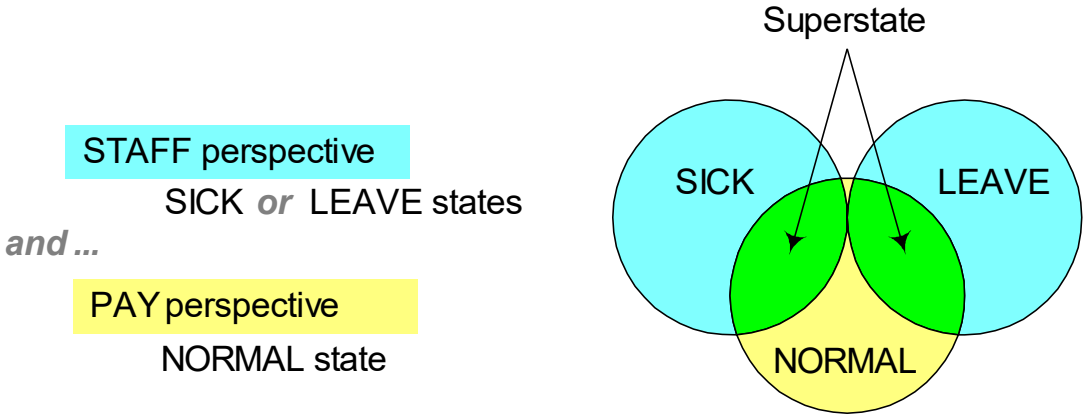


Figure -5: Superstate Example

Because SICK and LEAVE belong to the same perspective, either state will be included in the tally—but only where the employee’s time is also NORMAL, since that state belongs to a different perspective.

In addition to filtering schedule information by perspective state, a superstate can also filter schedule information by the following criteria:

- Day-of-week – by default, a superstate applies to all days of week, but a superstate can be defined to apply only to a subset of days of week. The day-of-week of a segment (and corresponding superstate

resolution) is based on the nominal date of that segment. If a superstate is defined to match only Monday, Wednesday, and Friday, then the same schedule information that resolves to the superstate on a nominal date of Monday, Wednesday, or Friday will not resolve to the superstate on other days of the week.

- Time-of-day – by default, a superstate applies to all times of day, but a superstate can be defined to apply only during certain times of day. The time-of-day of a segment (and corresponding superstate resolution) is based on the moment range of that segment. The time-of-day is calculated based on a time zone associated with the superstate definition. The time zone associated with a superstate can either be a specific time zone (identified by surrogate key) or the time zone of the employee associated with the schedule information. Any resolved spans will be limited based on the time-of-day filtering configured. Any superstate time that falls within the time-of-day filters will be counted in the superstate. Any superstate time that falls outside of the time-of-day filters will be ignored. The choice of time zone depends on the purpose of the superstate. If the superstate measures something about the employee's subjective experience, the employee's time zone should be used. For example, you could define a superstate that counted employee overtime during a cross-midnight schedules (11:00 PM to 1:00 AM in the employee's time zone). If the superstate is more of an objective measure, a specific time zone may be appropriate. For example, say 10:00 AM to 2:00 PM US Eastern time is a period of peak activity for your business. In that case, you could measure activities during this peak time with one or more superstates that use a time filter of 10:00 AM to 2:00 PM with a specific time zone of US Eastern time.
- Duration – by default, a superstate applies to all durations, but a superstate can be defined to apply only if the duration of the resolved span meets specified duration criteria:
- Duration is less than or equal to a specified upper bound.
- Duration is greater than or equal to a specified lower bound.
- Duration falls within a specified range.
- Resolved spans that match the specified duration criteria will be counted in the superstate. Resolved spans that fall outside of the specified duration criteria will be ignored.

Superstate filtering criteria are applied in the following order:

1. Perspective-state filtering. First, Tally Server resolves the segments of the specified employees in the perspectives associated with the superstate. It calculates the spans of time for which the schedule information resolves to at least one listed state in each of the associated perspectives. These resolved spans each have an associated nominal date and a range of moments.

2. Day-of-week filtering. If the superstate limits by day-of-week, compare the nominal dates of resolved spans to the day-of-week filter. Any spans whose nominal date falls on a day-of-week outside of the filter will be discarded.

3. Time-of-day filtering. If the superstate defines any time-of-day filters, calculate the time-of-day range for the remaining resolved spans using the time zone associated with the superstate. Remove portions of the resolved spans that fall outside of all time-of-day filters defined for the superstate. For example, say that the superstate applies during 9:00 AM-11:00 AM and 1:00 PM-3:00 PM in the employee's time zone. If the employee's schedule resolves to the superstate during a span of 10:00 AM-2:15 PM in the employee's time zone, then the resulting resolved spans will be 10:00 AM-11:00 AM and 1:00 PM-2:15 PM.

4. Duration filtering. If the superstate defines any duration filters, discard any remaining resolved spans that do not match any of the duration filters defined. For example, say that the superstate applies to resolved spans with durations greater than or equal to 1 hour. If the employee's schedule resolves to the superstate during the following spans: 8:00 AM-8:59 AM, 9:00 AM-10:00 AM, 11:00 AM-2:00 PM, the 8:00 AM-8:59 AM span will be discarded and the 9:00 AM-10:00 AM and 11:00 AM-2:00 PM spans will be kept.

Note: day-of-week filtering, time-of-day filtering, and duration filtering were introduced in version 22.0. Any Tally Server interfaces that depend on superstate definitions (ITCSMomentBasedTimePeriodTallyProvider, ITCSNomDateBasedScheduleListTallyProvider, ITCSNomDateBasedScheduleTallyProvider, and ITCSMomentBasedScheduleListTallyProvider (new in 22.0)) have been updated to support day-of-week filtering, time-of-day filtering, and duration filtering. As such, in 22.0 and higher versions, you will need to supply day-of-week, time-of-day, and duration filter information with your superstate definitions. You should update your application to read this information from the corresponding new tables (SPRSTATE_EXCL_DAYS, SPRSTATE_TIME_FLT, and SPRSTATE_DUR_FLT).

UTC Time

All times in Aspect Workforce are stored as some representation of Universal Coordinated Time (UTC), which was previously referred to as Greenwich Mean Time or GMT. By exclusively storing UTC times in the database it is possible to store the start and stop times for segments assigned to employees working in several different time zones in the same table, without regard to those time zones. When a user enters a time in the Aspect Workforce client it is converted to UTC using the local time zone setting of the user before being stored in the database. These times are then converted back to the local time for display to the user. Two units of time stored in the Aspect Workforce database relate to segments and schedules: Moments and Nominal Dates.

Moments

All start and stop times for segments in Aspect Workforce are stored as “moments”. A moment is defined to be the UTC time represented as the number of minutes since 12:00 AM December 30 1899. Therefore, the unit of moments is exactly one minute. A moment is the number of minutes since 12:00 AM UTC on December 30, 1899. The definition also allows for negative values for times that are earlier than that time, but these values are never used. Moments have a granularity of one minute. It is most important to note that moments specify an instant in time that is independent of time zones, i.e. the time specified by a moment does not change as the time zone changes. They are represented by a 32-bit integer. A duration is the difference between two moments.

An absolute moment range is represented by a start and a stop moment. The stop moment must be after the start moment. A relative moment range is represented by a start and a stop duration. The two durations are indexed to the current moment. The start duration must be less than the stop duration. A moment can be derived from a COleDateTime data type by converting it to minutes with the following formula:

```
moment = COleDateTime * 1440 (where 1440=24hours*60 minutes).
```

Fine Moments

Start and stop times that are expressed with a one-second granularity, e.g., sign-in times and sign-out times in Agent Productivity, are stored as “fine moments.” A fine moment is defined to be the UTC time represented as the number of seconds since 12:00 AM December 30 1899. Therefore, the unit of fine moments is exactly one second. A fine moment is the number of seconds since 12:00 AM UTC on December 30, 1899. The definition also allows for negative values for times that are earlier than that time, but these values are never used. Fine moments have a granularity of one second. It is most important to note that fine moments specify an instant in time that is independent of time zones, i.e. the time specified by a fine moment does not change as the time zone changes. They are represented by a whole number stored in a double-precision floating-point value. A duration is the difference between two fine moments.

An absolute fine moment range is represented by a start and a stop fine moment. The stop fine moment must be after the start fine moment. A fine moment can be derived from a COleDateTime data type by converting it to seconds with the following formula:

```
fine moment = COleDateTime * 86400 (where 86400=24hours*60 minutes*60
seconds).
```

Nominal Dates

The nominal date of a segment is the date that a segment is associated with for reporting and selection purposes. Nominal dates are represented as the number of days since December 30, 1899. Therefore, the unit of nominal dates is exactly one day. A nominal date is the number of days since December 30, 1899 (see COleDateTime). Nominal dates have a granularity of one day. Nominal dates are used in schedules as a way of grouping schedule segments together. The meaning of a nominal date is not as precise as the meaning of a moment because schedules can cross midnight and the date for a moment can change from one time zone to another. Nominal dates should be thought of as labels that are related to the start and stop moments of schedule segments, but are not fixed by them. The Updater service is responsible for enforcing the constraints on nominal dates and moments in official schedules. There are no constraints between moments and nominal dates in trial schedules. A nominal date is represented by a 32-bit integer.

An absolute nominal date range is represented by a start and a stop nominal date. The stop nominal date must be on or after the stop nominal date. There is no relative nominal date range in Tally Server.

Employee Groups

Employee groups are entities that employees can be assigned to for one or more range of nominal dates. Employee groups are arranged into hierarchical trees. Within a tree, an employee can be assigned to only one node on any given nominal date. Typical uses of employee groups would be to have trees that divided employees by geographical region or by categories such as full and part-time.

In Tally Server, an employee is assigned to an employee group if he is assigned to that node or any of the children of that node. The nominal date ranges of the assignment are given by a union of all of the ranges for that node and its children.

A list of one or more employee groups implies a list of employee group assignments. The following algorithm determines the list of assignments:

1. Group the list of employee groups by the employee group trees to which they belong.

2. Within each employee group tree, the list of assignments is given by a union of all of the assignments of that tree.

3. If there is more than one employee group tree, an intersection of the assignments within each tree is formed.

4. Employees with no nominal date ranges after step 3 are removed from the result.

Staff Groups

Staff Groups are used in forecasting and performance monitoring. They are arranged in hierarchical trees. A staff group may be associated with an employee group. A staff group tree is also associated with a state in the “staffing” perspective. Associating a perspective with the STF purpose designates the “staffing” perspective.

In Tally Server, employee groups are associated with a staff group if either that staff group or any of its children are associated with that employee group. Thus, a staff group may imply zero or more employee groups.

Data Types and Stored Procedures

This section discusses Alvaira Workforce data types and stored procedures that are relevant to the Tally Server API.

About Data Types

Aspect Workforce defines and uses several data types throughout its schema. Some of these data types are defined below. Note that these data types may be defined slightly differently on the different databases that Aspect Workforce supports. Also note that the names of all objects in some databases, such as Oracle, will be prefixed by the owner/creator of the schema, which is typically TCSDBOWNER.

Refer to the Aspect Workforce public data model for the most up to date information on all the tables in the Aspect Workforce schema. (For more information, see the Aspect Workforce Public Data Model Guide, available on the Alvaria Product Document Library.) The public data model also contains information about the primary, alternate, and foreign keys in each of the tables, and the relationships between tables.

Table 2: Data Types

Data Type	Domain	Description
SK	NUMERIC(15,1)	Surrogate Key: Used for the primary key of most tables. The SK is generated using either the GET_SK or the GET_SKS stored procedures to guarantee unique numbers.
NOM_DATE	NUMERIC(15,1)	Nominal Date: Used with both General and Detail segments to denote the day for which the schedule is in effect.
BOOLEAN	NCHAR(1)	Can be either ‘T’ or ‘F’, for true or false.
MEMO	BLOB SUB_TYPE 1 SEGMENT SIZE 80	Memo field: Used for entering memos for segments and other records. Columns with the data type of *Memo* are implemented as NCLOB in Oracle and nvarchar(max) in SQL Server.
START_MOMENT	NUMERIC(15,1)	Start Moment: Used to define the start moment for a segment.
STOP_MOMENT	NUMERIC(15,1)	Stop Moment: Used to define the stop moment for a segment.

Data Type	Domain	Description
TCS_MOMENT	NUMERIC(15,1)	Moment: Used to indicate a moment.
CODE	NVARCHAR(30)	Code: Used to indicate a character-based code. This is used as the data type for several fields, including segment code and segment category.
DESCRIPTION	NVARCHAR(30)	Description: Used for the descriptions of fields, usually those defined by type CODE.
TCS_POS_INTEGER	NUMERIC(15,1)	Positive integer: Used to define a field that must have a positive integer.
COLOR	NUMERIC(15,1)	Color: Used for the display of segments.
FONT_COLOR	NUMERIC(15,1)	Font Color: Used for the display of segments.
BMP	BLOB SUB_TYPE 0 SEGMENT_SIZE 80	Bitmap field: Used to define the bitmap image associated with a perspective state. Columns with the data type *Bmp* are implemented as BLOB in Oracle and varbinary(max) in SQL Server.

About Stored Procedures and Surrogate Keys

The Aspect Workforce data model defines several stored procedures. However, only the ones that you will need to use when using the Tally Server API are described in the following section.

Most of the primary keys in the Aspect Workforce database are surrogate keys. These keys must be unique in each table. There are two stored procedures that must be used to generate the surrogate keys used in Aspect Workforce:

GET_SK

The GET_SK procedure is used to generate a single surrogate key. It has one output parameter, SK_OUT (of type SK), which returns the generated SK.

GET_SKS

The GET_SKS procedure is used to generate a number of surrogate keys. This function is useful and faster if you need to generate a larger number of keys. It has one input parameter, NUM_SKS_IN, that specifies the number of keys to generate; it has one output parameter, SK_OUT, which is the first key in a continuous series of keys reserved for the process that called GET_SKS.

Database Tables

This section defines several database tables that are relevant to the Tally Server API.

Official Segment Tables

This section describes official segment tables that are relevant to the Tally Server API.

EMP

The EMP (Employee Records) table is an Official Segment table. This is the master table containing a list of all the employees in your Aspect Workforce system.

Table 3: EMP (Employee Records) Table

Column Name	Data Type	Description
EMP_SK	SK	Employee Surrogate Key
ID	NCHAR(10)	Employee ID (entered by user)
LAST_NAME	NVARCHAR(30)	Employee last name
FIRST_NAME	NVARCHAR(30)	Employee first name
EFF_HIRE_NOM_DATE	NOM_DATE	Employee effective-hire nominal date
TERM_NOM_DATE	NOM_DATE	Employee termination nominal date
ACTIVE_FLAG	BOOLEAN	Flag to indicate whether the employee is active or not
SHORT_NAME	NVARCHAR(30)	Abbreviated form of employee's name. Used by the Aspect Workforce client for display and in reports
SORT_NAME	NVARCHAR(60)	Alternate name for employee used by the Aspect Workforce client during sorting
TIME_ZONE_SK	SK	Surrogate key for time zone that the employee works in
SENIORITY	NVARCHAR(12)	Employee seniority
EMAIL_ADR	NVARCHAR(255)	Employee email address
MEMO	MEMO	Additional, unstructured information stored for this employee
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record.

Column Name	Data Type	Description
UPD_TS	DATE	Date and time at which the associated record was updated.

GEN_SEG

The GEN_SEG (General Segments) table is an Official Segments table. It contains all the official general segments in your Aspect Workforce system.

Table-4: GEN_SEG (General Segments) Table

Column Name	Data Type	Description
GEN_SEG_SK	SK	General Segment surrogate key
SEG_CODE_SK	SK	Segment Code surrogate key (segment code for general segment)
EMP_SK	SK	Employee surrogate key (employee assigned to general segment)
NOM_DATE	NOM_DATE	Nominal Date for general segment
MEMO	MEMO	Memo associated with general segment

DET_SEG

The DET_SEG (Detail Segments) table is an Official Segments table. It contains all the official detail segments in your Aspect Workforce system.

Table -5: DET_SEG (Detail Segments) Table

Column Name	Data Type	Description
DET_SEG_SK	SK	Detail Segment surrogate key
SEG_CODE_SK	SK	Segment Code surrogate key (segment code for detail segment)
EMP_SK	SK	Employee surrogate key (employee assigned to detail segment)
NOM_DATE	NOM_DATE	Nominal Date for detail segment
START_MOMENT	START_MOMENT	Start Moment for Detail Segment
STOP_MOMENT	STOP_MOMENT	Stop Moment for Detail Segment
MEMO	MEMO	Memo associated with general segment

DLY_SEG_SMRY

The DLY_SEG_SMRY (Daily Segment Summary) table is an Official Segments table. It contains summaries of all the official segments on a per-employee per-nominal date basis.

Table -6: DLY_SEG_SMRY (Daily Segment Summary) Table

Column Name	Data Type	Description
NOM_DATE	NOMINAL_DATE	Nominal Date for Daily Segment summary
EMP_SK	SK	Employee for Daily Segment Summary
EARL_START_MOMENT	TCS_MOMENT	Earliest Start Moment for employee for Nominal Date
LATE_STOP_MOMENT	TCS_MOMENT	Latest Stop Moment for employee for Nominal Date
GEN_SEG_CODE_SK	SK	Highest ranking General Segment for Nominal Date
WORK_SEG_CODE_SK	SK	Highest ranking work segment

Segment Information Tables

This section describes segment information tables that are relevant to the Tally Server API.

SEG_CODE

The SEG_CODE (Segment Code) table is a Segment Information table. It contains the segment code definitions used in creating general and detail segments.

Table -7: SEG_CODE (Segment Code) Table

Column Name	Data Type	Description
SEG_CODE_SK	SK	Segment code surrogate key
CODE	CODE	Character representation of Segment Code. This is what is entered by users when creating segments
RANK	TCS_POS_INTEGER	The rank ordering of segment, used in resolving overlapping segments
DESCR	DESCRIPTION	Brief description of segment code
REQ_MEMO	BOOLEAN	Determines whether a memo is required when entering a segment with this code

Column Name	Data Type	Description
COLOR	COLOR	Background color when displaying the segment code
FONT_COLOR	FONT_COLOR	Font color for the segment code letters when displayed.
DEF_DUR	TCS_POS_INTEGER	Default duration (in minutes) of a segment. Must be between 1 and 1439.
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record.
UPD_TS	DATE	Date and time at which the associated record was updated.

SEG_CAT

The SEG_CAT (Segment Category) table is a Segment Information table. It contains the definitions of segment categories that can be used to define segment entry rules.

Table -8: SEG_CAT (Segment Category) Table

Column Name	Data Type	Description
SEG_CAT_SK	SK	Segment Category surrogate key.
CODE	CODE	Segment Category code.
DESCR	DESCRIPTION	Description if segment category.
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record.
UPD_TS	DATE	Date and time at which the associated record was updated.

SEG_CODE_CAT_ASSOC

The SEG_CODE_CAT_ASSOC (Code/Category Association) table is a Segment Information table. It contains the associations between the segment categories and segment codes.

Table -9: SEG_CODE_CAT_ASSOC (Code/Category Association) Table

Column Name	Data Type	Description
SEG_CAT_SK	SK	Segment Category surrogate key
SEG_CODE_SK	SK	Segment Code surrogate key

Other Tables

This sections describes tables relevant to the Tally Server API that are not official segment tables or segment information tables.

PERSP

The PERSP table is a Perspective table. It contains the associations between perspective codes.

Table -10: PERSP (Perspective) Table

Column Name	Data Type	Comments
PERSP_SK	SK	Perspective surrogate key.
CODE	CODE	Perspective code.
DESCR	DESCRIPTION	Description of perspective.
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record.
UPD_TS	DATE	Date and time at which the associated record was updated.

STATE

The STATE table shows a state with a perspective.

Table -11: STATE Table

Column Name	Data Type	Comments
STATE_SK	SK	State surrogate key
PERSP_SK	SK	Perspective surrogate key
CODE	CODE	State code
DESCR	DESCRIPTION	State description
STATE_ICON_SK	SK	State icon surrogate key
COLOR	COLOR	Background color when displaying the segment code
FONT_COLOR	FONT_COLOR	Font color for the segment code letters when displayed.
BMP	BMP	The bitmap for the state icon.

SEG_STATE_ASSOC

The SEG_STATE_ASSOC (Segment Code to State Association) table contains the associations between the segment categories and segment codes.

Table -12: SEG_STATE_ASSOC (Segment Code to State Association) Table

Column Name	Data Type	Comments
SEG_CODE_SK	SK	Segment code surrogate key
STATE_SK	SK	State surrogate key

SPRSTATE_MBR

The SPRSTATE_MBR (Superstate State Association) table contains the associations between the superstates and state codes.

Table -13: SPRSTATE_MBR (Superstate State Association) Table

Column Name	Data Type	Comments
SPRSTATE_SK	SK	Superstate surrogate key
STATE_SK	SK	State surrogate key

SPRSTATE

The SPRSTATE (Superstate) table is the master table containing a list of all superstates.

Table -14: SPRSTATE (Superstate) Table

Column Name	Data Type	Comments
SPRSTATE_SK	SK	Superstate surrogate key
CODE	CODE	Superstate code
DESCR	DESCRIPTION	Description of superstate
HRS_UPD_FLAG	CHAR(1)	Hours Update Flag. Indicates whether or not Updater should store information on this superstate in SPRSTATE_HRS; once the flag has been set to T, it cannot be changed to F. Valid values: T = Updater stores information on this superstate in SPRST
TIME_ZONE_SK	SK	If this superstate defines one or more time-of-day filters based on a specific time zone, this column will contain the surrogate key of that specific time zone. Otherwise, if the superstate defines one or more time-of-day filters based on the

Column Name	Data Type	Comments
		employee's time zone or the superstate does not define any time-of-day filters, this column will be NULL.
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record.
UPD_TS	DATE	Date and time at which the associated record was updated.

SPRSTATE_EXCL_DAYS

The SPRSTATE_EXCL_DAYS (Superstate exclude days) table is used to define the day-of-week filter for a superstate. It lists any days of week that should be excluded from the superstate.

Table -15: SPRSTATE_EXCL_DAYS (Superstate Exclude Days) Table

Column Name	Data Type	Comments
SPRSTATE_SK	SK	Superstate surrogate key
DOW	NUMBER(15,1)	0=Sunday, 1=Monday, etc. A superstate that applies on all days of week will have no rows in this table.

SPRSTATE_TIME_FLT

The SPRSTATE_TIME_FLT (Superstate time filters) table is used to define the time-of-day filters for a superstate.

Table -16: SPRSTATE_TIME_FLT (Superstate Time Filters) Table

Column Name	Data Type	Comments
SPRSTATE_TIME_FLT_SK	SK	Superstate time filter surrogate key
SPRSTATE_SK	SK	Superstate surrogate key
START_MI	NUMBER(15,1)	0 to 1439. 0=Midnight, 1=12:01 AM, etc.
STOP_MI	NUMBER(15,1)	0 to1439. 0=Midnight, 1=12:01 AM, etc.

SPRSTATE_DUR_FLT

The SPRSTATE_DUR_FLT (Superstate duration filters) table is used to define the duration filters for a superstate.

Table -17: SPRSTATE_DUR_FLT (Superstate Duration Filters) Table

Column Name	Data Type	Comments
SPRSTATE_DUR_FLT_SK	SK	Superstate duration filter surrogate key
SPRSTATE_SK	SK	Superstate surrogate key
MIN_MI	NUMBER(15,1)	0 to 5760. The lower bound of the duration filter, in minutes.
MAX_MI	NUMBER(15,1)	0 to 5760. The upper bound of the duration filter, in minutes.

SPRSTATE_HRS

The SPRSTATE_HRS (Superstate Hours Entity Name) table stores superstate hours by employee and nominal date.

Table-18: SPRSTATE_HRS (Superstate Hours Entity Name) Table

Column Name	Data Type	Comments
SPRSTATE_SK	NUMBER(15,1)	Superstate surrogate key
EMP_SK	NUMBER(15,1)	Employee surrogate key
NOM_DATE	NUMBER(15,1)	Nominal date
MI	NUMBER(15,1)	The number of minutes the employee was in this superstate for this nominal date Valid values: 0 to 5760
UPD_TS	DATE	Time at which this record was updated

Tally Server Architecture

The Tally Server server is designed in a multi-tier architecture. The Aspect Workforce clients and any other applications that want to query for state information with respect to an employees' official or trial schedule cannot get resolved schedule information by directly querying the database. Instead, they communicate with the Tally Server server, which enforces the segment priorities set up in the Aspect Workforce system to ensure that data integrity is maintained and the customer's segment entry rules are enforced. Clients of the Tally Server server, including the Aspect Workforce client, that want to request official schedules communicate with the Tally Server server using the Tally Transport in-process Com server, TallyTransport.dll. Tally Transport provides a common interface for all clients of Tally Server, and handles the communication with Tally Server.

The Tally Server server does not reject any valid request. This means that a client's request can potentially cause performance issues with other requests that it is currently working on. It is the responsibility of the client to ensure that its requests are appropriate for the WFM environment.

Tally Server Server

The Tally Server server is a Windows system service that is designed to handle concurrent requests from many Tally Server clients. The Tally Server server queues every request it receives and performs validation on them in the order they are received. When it is finished evaluating a request, it provides a response back to the client. Tally Server can communicate with a Tally Transport by one of two methods: a direct connection via DCOM or a message passed by the Alvaria Message Routing server. Alvaria Message Routing is used in load balanced configurations. The WFM Installation Guide and WFM System Administration Guide have more information on this topic.

Tally Transport

Tally Transport is installed on all WFM workstations and application servers. It provides a common COM interface for any client that connects to the Tally Server server. Tally Transport can send requests to Tally Server using either DCOM or the Alvaria Message Routing Service, depending upon the configuration of WFM. The repackaging of requests sent to Tally Server and respsonses received from Tally Server does not include any significant data validation; therefore, it is important for the client to verify all data that is sent to the server.

Tally Server Concepts

In order to request a schedule the Segment selector must be built to tell Tally Server what information to retrieve. Tally Server returns a pointer to a resolution.

Resolution Request

The resolution request provides information to the Tally Server so that it can retrieve the desired schedule information. It is comprised of a Resolution definition, a list of perspectives or superstates depending on whether it is a request for schedule or a tally, and description information. The MomentBasedScheduleResolution definition is comprised of the segment selector and moment ranges, and a Merge Schedule Boolean (MergeAssignedSchedules) which the client uses to tell Tally Server to either treat every schedule selected as an individual entity or to merge all schedule assigned to the same employee into one super-schedule.

If the MergeAssignedSchedules flag is false, the resolution identifier will follow the same rules as schedule identifiers since there is only one schedule in each super-schedule.

If the MergeAssignedSchedules flag is true, any super-schedule that comes from one or more assigned schedules will have the employee key set to the same value as found in the EMP database table for that employee. The trial schedule and trial schedule set keys will both be set to BadKey. Unassigned trial schedules will set the employee key to BadKey and the trial schedule and trial schedule set keys will be set to the values from the TRL_SCH and SCH_SET database tables. (There is no such thing as an unassigned official schedule.)

```
typedef struct _TMomentBasedScheduleResolutionRequest
       {
       TMomentBasedScheduleResolutionDefinition Definition;
       TPerspectiveKeyList Perspectives;
       PTCSComString pAlias;
       PTCSComString pDescription;
       }TMomentBasedScheduleResolutionRequest;typedef struct

_TMomentBasedScheduleResolutionDefinition
       {
       TSegmentSelectorList SegmentSelectors;
       TAbsoluteMomentRangeList AbsoluteMomentRanges;
```

```
        TRelativeMomentRangeList RelativeMomentRanges;
        BOOL MergeAssignedSchedules;
        }TMomentBasedScheduleResolutionDefinition;

typedef struct  _TRelativeMomentRangeList
        {
        long NRanges;
        /* [unique][size_is] */ TRelativeMomentRange *pMomentRange;
        }TRelativeMomentRangeList;typedef struct  _TRelativeMomentRange
        {
        TDuration Start;
        TDuration Stop;
        }TRelativeMomentRange;typedef struct  _TPerspectiveKeyList
        {
        long NKeys;
        /* [unique][size_is] */ TPerspectiveKey *pSK;
        }TPerspectiveKeyList;
```

Segment Selector

Segment Selectors are a common part of all Tally Server requests. They are used to determine what schedules (both official and trial) are to be read from the database. The request will be for either a list of employees, a group of employees, or a staff group. If an official schedule is desired, set the UseOfficialSchedules Boolean to true, or else provide the trial schedule information. A segment selector contains the following information:

- A list of employees
- A list of employee groups
- A list of staff groups
- A list of trial schedules
- A list of trial schedule sets
- A flag to indicate whether or not to look at official schedules
- A flag to indicate whether or not to select all schedules that have been assigned to an employee
- A flag to indicate whether or not to select all schedules that have not been assigned to an employee.

```
typedef struct  _TSegmentSelector
        {
        BOOL SelectAllAssignedSchedules;
        BOOL SelectAllUnassignedSchedules;
        TEmployeeKeyList Employees;
        TEmployeeGroupKeyList EmployeeGroups;
        TStaffGroupKeyList StaffGroups;
        TTrialScheduleKeyList TrialSchedules;
        TTrialScheduleSetKeyList TrialScheduleSets;
        BOOL UseOfficialSchedules;
        }TSegmentSelector;typedef struct  _TTrialScheduleKeyList
        {
        long NKeys;
        /* [unique][size_is] */ TTrialScheduleKey *pSK;
        }TTrialScheduleKeyList;typedef struct  _TEmployeeGroupKeyList
        {
        long NKeys;
        /* [unique][size_is] */ TEmployeeGroupKey *pSK;
        }TEmployeeGroupKeyList;typedef struct  _TEmployeeKeyList
        {
        long NKeys;
        /* [unique][size_is] */ TEmployeeKey *pSK;
```

```
}TEmployeeKeyList;
```

Schedule and Resolution Identifier

The resolution identifier is used to identify the super-schedules for the information returned from Tally Server to the client. (It is not used for the moment-based time period tally which sums over super-schedules.) The resolution identifier looks like the schedule identifier, but it has different semantics. Internally, Tally Server uses a schedule identifier to identify every schedule (both official and trial). A schedule identifier contains three items: an employee key, a trial schedule key, and a trial schedule set key. If the schedule is assigned to an employee, the employee key will match the primary key of the record in the EMP database table for that employee. If the schedule is unassigned, the value of the employee key is BadKey. If the schedule is an official schedule, both the trial schedule and the trial schedule set keys will be BadKey. If the schedule is a trial schedule, the trial schedule and trial schedule set keys will contain the values of the primary keys from the TRL_SCH and SCH_SET database tables.

```
typedef struct  _TFilteredScheduleResolution
        {
        TResolutionIdentifier ResolutionIdentifier;
        long NPerspectives;
        /* [size_is] */ TTimePeriodResolutionForPerspectiveList
        *pTimePeriodResolutionForPerspectiveList;
        }TFilteredScheduleResolution;typedef struct
        _TResolutionIdentifier
        {
        TEmployeeKey EmployeeSK;
        TTrialScheduleKey TrialScheduleSK;
        TTrialScheduleSetKey TrialScheduleSetSK;
        }TResolutionIdentifier;
```

Internal Processing

Tally Server handles a request by first acquiring any needed schedule information from the database. The schedule information may be from both official and trial schedules. A set of super-schedules is then created from these schedules. A super-schedule is a combination of one or more schedules. Multiple schedules that are assigned to the same employee may be combined if the appropriate flag is set in the request sent to Tally Server. The super-schedule is then resolved to determine what state the schedule is in for each of the perspectives defined in the database. This super-schedule resolution can be queried internally by either moment ranges or nominal date ranges.

If the request is for a tally, the super-schedule resolution is evaluated by the superstates in the request to produce a set of super-schedule superstate resolutions. A superstate is a list of states from one or more perspectives that is used to construct a Boolean function that operates on the set of resolved states from a super-schedule resolution. This data consists of a set of time periods when applying the superstate to the super-schedule resolution evaluates to true. This super-schedule superstate resolution can be queried internally by either moment ranges or nominal date ranges.

Internally, Tally Server uses the list of segment selectors to build a map of objects indexed by the resolution identifiers. These objects contain the list of schedule identifiers implied by the segment selectors. They also contain a list of nominal date ranges for which schedule information is to be retrieved from the database. The algorithm is as follows:

1. Create an empty map of super-schedule resolver information objects.

2. Loop through the segment selectors. (No Aspect Workforce module has ever used more than one segment selector in a Tally Server request. Consequently, this has not been tested.)

3. Add all of the explicit trial schedules from the list of trial schedules to the map. The nominal date range for these trial schedules is the full range of nominal dates implied by the request. (This allows

one to include trial schedules that come from any trial schedule set. If any of these trial schedules are assigned, they may have the side effect of including more schedule information than implied by employee group assignments.)

4. If the SelectAllUnassignedSchedules flag is true, add all of the unassigned trial schedules from all of the trial schedule sets listed to the map. The nominal date range for these trial schedules is the full range of nominal dates implied by the request.

5. If the SelectAllAssignedSchedules flag is true, the lists of employees, employee groups, and staff groups will be ignored. Instead, schedules will be added as follows:

    a. Add all of the assigned trial schedules from all of the trial schedule sets listed to the map. The nominal date range for these trial schedules is the full range of nominal dates implied by the request.

    b. If the UseOfficialSchedules flag is true, add schedule identifiers for official schedules for all employees to the map. The nominal date range for these official schedules is the full range of nominal dates implied by the request. (This may have the side effect of including more schedule information than implied by employee group assignments.) (It is almost guaranteed that one will never want to do this, especially in a large call center.

6. If the SelectAllAssignedSchedules flag is false, Tally Server uses the lists of employees, employee groups, and staff groups to determine which schedules to include. They are added according to the following process:

    a. All employee groups implied by the listed staff groups are determined. These are added to the listed employee groups.

    b. The employee group assignments for all of these employee groups are determined.

    c. A virtual employee group with its own virtual employee groups tree is formed from the listed employees. The nominal date range for the assignment of these employees in the virtual group is from the beginning of time to the end of time.

    d. The employee assignments based on this calculated list of employee groups are done by the usual rules of unions within employee group trees and intersections between employee group trees.

7. The assignments determined in step 6 are used to determine which trial schedules in the listed trial schedules are needed. These trial schedules are added to the map of super-schedule resolver information objects. The nominal date ranges are those from the calculated assignments.

    a. If the UseOfficialSchedules flag is set to true, the map has entries added for all of the assignments from step 6.

Tally Server API Implementation

This section describes the Tally Server API that applications must use when submitting changes to the Tally Server server. The Tally Server API consists of several COM interfaces that are used to submit requests for schedule information.

Basic Programming Model

Connecting to Tally Server, sending it a request for information, and handling the response follows a typical pattern for all four request types. The pattern is as follows:

1. Use CoCreateInstance or CoCreateInstanceEx(DCOM) to connect to Tally Server. Get a pointer to the ITCSTallyServer interface.

2. Create the request structure and fill it with the request.

3. Use the appropriate Get routine to get a pointer to a provider that will be able to provide the results that you have requested.

4. If there are no more requests, release the ITCSTallyServer interface pointer. You may wish to do this step at a later point.

5. Use the Get routine from the interface pointer returned in step 3 to get the results of your request.57

6. If desired, copy the returned data structures into more convenient forms such as C++ objects with functions that provide access to the data. In any case, when you are finished with the returned structures, do not forget to use CoTaskMemFree to free the memory held by the returned structures. Remember that every non-zero pointer requires its own call to CoTaskMemFree.

7. If you are finished with the provider, release the interface pointer, otherwise go back to step 5 for new results.

There are a few common items that appear in every request. The pRequestor string is used internally to attach a name of the entity or user who made the request. This string is used in some of Tally Server's diagnostic output. There is also a description string embedded in every request. It, also, is only used for diagnostic purposes. Finally, there is a pAlias string. This string must match the BDE alias name of the database you want to use and from which all of the database keys you are using comes from. The value must match the string as it is found in the BDE configuration on the application server where the Tally Server you are using is running. The pJobHandle quantity returned by every Get routine in the ITCSTallyServer interface can be used for diagnostic purposes. So far, only Tally Server itself has used any of these diagnostics.

COM Interfaces Reference

The section describes the COM interfaces of the Tally Server API.

About the ITCSTallyServer Tally Server Interface

The ITCSTallyServer Tally Server interface consists of four primary methods. These methods are used to submit the request to Tally Server:

1. GetMomentBasedResolutionProvider

2. GetMomentBasedTimePeriodTallyProvider

3. GetNomDateBasedScheduleListTallyProvider

4. GetNomDateBasedScheduleTallyProvider,

The following table shows the associated interfaces that are used to get the resolved schedule information.

Table-19: Associated Tally Server Interfaces

Interface	Method
ITCSMomentBasedResolutionProvider	GetResolvedScheduleInfo
ITCSMomentBasedTimePeriodTallyProvider	GetTallyInfo
ITCSNomDateBasedScheduleListTallyProvider	GetTallyInfo
ITCSNomDateBasedScheduleTallyProvider	GetTallyInfo
ITCSMomentBasedScheduleListTallyProvider	GetTallyInfo

ITCSTallyServer Interface

ITCSTallyServer is an interface to four methods that submit requests to Tally Server.

GetMomentBasedResolutionProvider Method

```
HRESULT GetMomentBasedResolutionProvider ( [in] PCTCSComString
pRequestor, [in] TMomentBasedScheduleResolutionRequest* pRequest,

[out] ITCSMomentBasedResolutionProvider** ppProvider, [out]
TJobHandle* pJobHandle )
```

This method gets a moment-based time period tally provider for the moment based tally request.

A related interface, ITCSMomentBasedResolutionProvider Interface, gets the schedule information when given the pointer to the resolution provider.

Parameters

The following table describes the parameters of the GetMomentBasedResolutionProvider method.

Table-20: GetMomentBasedResolutionProvider Parameters

Parameter	Description
PCTCSComString pRequestor	String to identify source of the request
TMomentBasedScheduleResolutionRequest * pRequest	Pointer to the request structure (defined below)
ITCSMomentBasedResolutionProvider ** ppProvider	Pointer to the Resolution provider
TJobHandle* pJobHandle	Job Handle for debugging

Structure Definitions

The following are the structure definitions for the methods. In order to obtain the results the members of the appropriate structure need to be populated.

```
typedef struct TMomentBasedScheduleResolutionRequest
        {
```

```
TMomentBasedScheduleResolutionDefinition Definition;
TPerspectiveKeyList Perspectives;
PTCSComString pAlias;
PTCSComString pDescription;
}TMomentBasedScheduleResolutionRequest;
typedef struct TMomentBasedScheduleResolutionDefinition
{
TSegmentSelectorList SegmentSelectors;
TAbsoluteMomentRangeList AbsoluteMomentRanges;
TRelativeMomentRangeList RelativeMomentRanges;
BOOL MergeAssignedSchedules;
}TMomentBasedScheduleResolutionDefinition;
typedef struct TSegmentSelectorList
{
long NSelectors;
/* [unique][size_is] */ TSegmentSelector *pSegmentSelector;
}TSegmentSelectorList;
typedef struct _TSegmentSelector
{
BOOL SelectAllAssignedSchedules;
BOOL SelectAllUnassignedSchedules;
TEmployeeKeyList Employees;
TEmployeeGroupKeyList EmployeeGroups;
TStaffGroupKeyList StaffGroups;
TTrialScheduleKeyList TrialSchedules;
TTrialScheduleSetKeyList TrialScheduleSets;
BOOL UseOfficialSchedules;
}TSegmentSelector;
```



Note: :(See TSegmentSelector definition below)

```
typedef struct TAbsoluteMomentRangeList
{
long NRanges;
/* [unique][size_is] */ TAbsoluteMomentRange *pMomentRange;
}TAbsoluteMomentRangeList;
typedef struct TAbsoluteMomentRange
{
TMoment Start;
TMoment Stop;
}TAbsoluteMomentRange;
typedef struct TRelativeMomentRangeList
{
long NRanges;
[size_is ( NRanges ), unique] TRelativeMomentRange*
pMomentRange;
} TRelativeMomentRangeList;
typedef struct _TRelativeMomentRange
{
TDuration Start;
TDuration Stop;
} TRelativeMomentRange;
```

GetMomentBasedTimePeriodTallyProvider Method

```
HRESULT GetMomentBasedTimePeriodTallyProvider ( [in] PCTCSComString
pRequestor,[in] TMomentBasedTallyRequest* pRequest, [out]
ITCSMomentBasedTimePeriodTallyProvider** ppProvider, [out]
TJobHandle* pJobHandle );
```

The GetMomentBasedTimePeriodTallyProvider method gets a moment-based tally provider for the moment-based tally request.

A related interface, ITCSMomentBasedTimePeriodTallyProvider Interface, gets the schedule information when given the pointer to the tally provider.

Parameters

The following table describes the parameters of the GetMomentBasedTimePeriodTallyProvider method.

Table-21: GetMomentBasedTimePeriodTallyProvider Parameters

Parameter	Description
PCTCSComString pRequestor	String to identify source of the request
TMomentBasedTallyRequest * pRequest	Pointer to the request structure (defined below)
ITCSMomentBasedTimePeriodTallyProvider ** ppProvider	Pointer to the Resolution provider
TJobHandle* pJobHandle	Job Handle for debugging

Structure Definitions

The following are the structure definitions for the methods. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct _TMomentBasedTallyRequest
        {
        TMomentBasedScheduleResolutionDefinition Definition;
        TSuperStateWithIDList SuperStates;
        PTCSComString pAlias;
        PTCSComString pDescription;
        }TMomentBasedTallyRequest;
        typedef struct_TMomentBasedScheduleResolutionDefinition
        {
        TSegmentSelectorList SegmentSelectors;
        TAbsoluteMomentRangeList AbsoluteMomentRanges;
        TRelativeMomentRangeList RelativeMomentRanges;
        BOOL MergeAssignedSchedules;
        }TMomentBasedScheduleResolutionDefinition;
        typedef struct_TSuperStateWithIDList
        {
        long NItems;
        /* [size_is] */ TSuperStateWithID *pItem;
        }TSuperStateWithIDList;
        typedef struct TSuperStateWithID
        {
        TStateKeyList SuperState;
        TDayOfWeekMask Days;
        TTimeFilterList TimeFilters;
        TDurationFilterList DurationFilters;
        long ID;
        }TSuperStateWithID;
        typedef struct TStateKeyList
        {
        long NKeys;
        /* [unique][size_is] */ TStateKey *pSK;
        }TStateKeyList;
        typedef long TDayOfWeekMask;
        const long DOWSunday = 1;
```

```
const long DOWMonday = 2;
const long DOWTuesday = 4;
const long DOWWednesday = 8;
const long DOWThursday = 16;
const long DOWFriday = 32;
const long DOWSaturday = 64;
const TDayOfWeekMask DOWAllDays = DOWSunday + DOWMonday +
  DOWTuesday + DOWWednesday +
  DOWThursday + DOWFriday +
  DOWSaturday;
typedef struct _TTimeFilterList
{
TTimeZoneKey TZKey;
long NFilters;
[size_is(NFilters)] TTimeFilter* pFilters;
} TTimeFilterList;
typedef struct _TTimeFilter
{
TTimeOfDay Start;
TTimeOfDay Stop;
} TTimeFilter;
typedef long TTimeOfDay;// Minutes since midnight
typedef struct _TDurationFilterList
{
long NFilters;
[size_is(NFilters)] TDurationFilter* pFilters;
} TDurationFilterList;
typedef struct _TDurationFilter
{
TDuration Min;
TDuration Max;
} TDurationFilter;
typedef long TDuration;
typedef struct TSegmentSelectorList
{
long NSelectors;
/* [unique][size_is] */ TSegmentSelector *pSegmentSelector;
}TSegmentSelectorList;
typedef struct_TSegmentSelector
{
BOOL SelectAllAssignedSchedules;
BOOL SelectAllUnassignedSchedules;
TEmployeeKeyList Employees;
TEmployeeGroupKeyList EmployeeGroups;
TStaffGroupKeyList StaffGroups;
TTrialScheduleKeyList TrialSchedules;
TTrialScheduleSetKeyList TrialScheduleSets;
BOOL UseOfficialSchedules;
}TSegmentSelector;
```

Note: See TSegmentSelector definition below.

```
typedef struct TAbsoluteMomentRangeList
{
long NRanges;
/* [unique][size_is] */ TAbsoluteMomentRange *pMomentRange;
}TAbsoluteMomentRangeList;
typedef struct_TAbsoluteMomentRange
{
TMoment Start;
TMoment Stop;
}TAbsoluteMomentRange;
```

```
typedef struct_TRelativeMomentRangeList
{
long NRanges;
[size_is ( NRanges ), unique] TRelativeMomentRange*
pMomentRange;
} TRelativeMomentRangeList;
typedef struct _TRelativeMomentRange
{
TDuration Start;
TDuration Stop;
} TRelativeMomentRange;
```

GetNomDateBasedScheduleListTallyProvider Method

```
HRESULT GetNomDateBasedScheduleListTallyProvider ( [in]
PCTCSComString pRequestor,[in] TNomDateBasedTallyRequest* pRequest,
[out] ITCSNomDateBasedScheduleListTallyProvider** ppProvider,[out]
TJobHandle* pJobHandle )
```

The GetNomDateBasedScheduleListTallyProvider method gets a nominal-date-based schedule tally provider for the nominal-date-based tally request.

A related interface, ITCSNomDateBasedScheduleListTallyProvider Interface, gets the schedule information when given the pointer to the resolution provider.

Parameters

The following table describes the parameters of the GetNomDateBasedScheduleListTallyProvider method.

Table -22: GetNomDateBasedScheduleListTallyProvider Parameters

Parameter	Description
PCTCSComString pRequestor	String to identify source of the request
TNomDateBasedTallyRequest* pRequest	Pointer to the request
ITCSNomDateBasedScheduleListTallyProvider** ppProvider	Pointer to the Tally provider
TJobHandle* pJobHandle	Job Handle for debugging

Structure Definitions

The following are the structure definitions for the methods. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct_TNomDateBasedTallyRequest
       {
       TNomDateBasedScheduleResolutionDefinition Definition;
       TSuperStateWithIDList SuperStates;
       PTCSComString pAlias;
       PTCSComString pDescription;
       }TNomDateBasedTallyRequest;
       typedef struct_TNomDateBasedScheduleResolutionDefinition
       {
       TSegmentSelectorList SegmentSelectors;
       TAbsoluteNomDateRangeList AbsoluteNomDateRanges;
       BOOL MergeAssignedSchedules;
       }TNomDateBasedScheduleResolutionDefinition;
```

```
typedef struct_TSuperStateWithIDList
{
long NItems;
/* [size_is] */ TSuperStateWithID *pItem;
}TSuperStateWithIDList;
typedef struct TSuperStateWithID
{
TStateKeyList SuperState;
TDayOfWeekMask Days;
TTimeFilterList TimeFilters;
TDurationFilterList DurationFilters;
long ID;
}TSuperStateWithID;
typedef struct TStateKeyList
{
long NKeys;
/* [unique][size_is] */ TStateKey *pSK;
}TStateKeyList;
typedef long TDayOfWeekMask;
const long DOWSunday = 1;
const long DOWMonday = 2;
const long DOWTuesday = 4;
const long DOWWednesday = 8;
const long DOWThursday = 16;
const long DOWFriday = 32;
const long DOWSaturday = 64;
const TDayOfWeekMask DOWAllDays = DOWSunday + DOWMonday +
DOWTuesday + DOWWednesday +
DOWThursday + DOWFriday +
DOWSaturday;
typedef struct _TTimeFilterList
{
TTimeZoneKey TZKey;
long NFilters;
[size_is(NFilters)] TTimeFilter* pFilters;
} TTimeFilterList;
typedef struct _TTimeFilter
{
TTimeOfDay Start;
TTimeOfDay Stop;
} TTimeFilter;
typedef long TTimeOfDay;// Minutes since midnight
typedef struct _TDurationFilterList
{
long NFilters;
[size_is(NFilters)] TDurationFilter* pFilters;
} TDurationFilterList;
typedef struct _TDurationFilter
{
TDuration Min;
TDuration Max;
} TDurationFilter;
typedef long TDuration;
typedef struct_TSegmentSelectorList
{
long NSelectors;
/* [unique][size_is] */ TSegmentSelector *pSegmentSelector;
}TSegmentSelectorList;
typedef struct_TSegmentSelector
{
BOOL SelectAllAssignedSchedules;
BOOL SelectAllUnassignedSchedules;
TEmployeeKeyList Employees;
```

```
TEmployeeGroupKeyList EmployeeGroups;
TStaffGroupKeyList StaffGroups;
TTrialScheduleKeyList TrialSchedules;
TTrialScheduleSetKeyList TrialScheduleSets;
BOOL UseOfficialSchedules;
}TSegmentSelector;
Note:See TSegmentSelector definition below.
typedef struct _TAbsoluteNomDateRangeList
{
long NRanges;
/* [unique][size_is] */ TAbsoluteNomDateRange *pNomDateRange;
}TAbsoluteNomDateRangeList;
typedef struct TAbsoluteNomDateRange
{
TNomDate Start;
TNomDate Stop;
}TAbsoluteNomDateRange;
```

GetNomDateBasedScheduleTallyProvider Method

```
HRESULT GetNomDateBasedScheduleTallyProvider ( [in] PCTCSComString
pRequestor, [in] TNomDateBasedTallyRequest* pRequest, [out]
ITCSNomDateBasedScheduleTallyProvider** ppProvider, [out] TJobHandle*
pJobHandle );
```

The GetNomDateBasedScheduleTallyProvider method gets a nominal-date-based schedule tally provider for the nominal date based tally request.

A related interface, ITCSNomDateBasedScheduleTallyProvider Interface, gets the schedule information given the pointer to the resolution provider.

Parameters

The following table describes the parameters of the GetNomDateBasedScheduleTallyProvider method.

Table -23: GetNomDateBasedScheduleTallyProvider Parameters

Parameter	Description
PCTCSComString pRequestor	String to identify source of the request
TNomDateBasedTallyRequest* pRequest	Pointer to the request
ITCSNomDateBasedScheduleTallyProvider** ppProvider	Pointer to the Tally provider
TJobHandle* pJobHandle	Job Handle for debugging

Structure Definitions

The following are the structure definitions for this method. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct_TNomDateBasedTallyRequest
      {
      TNomDateBasedScheduleResolutionDefinition Definition;
      TSuperStateWithIDList SuperStates;
      PTCSComString pAlias;
      PTCSComString pDescription;
      }TNomDateBasedTallyRequest;
      typedef struct_TNomDateBasedScheduleResolutionDefinition
```

```
{
TSegmentSelectorList SegmentSelectors;
TAbsoluteNomDateRangeList AbsoluteNomDateRanges;
BOOL MergeAssignedSchedules;
}TNomDateBasedScheduleResolutionDefinition;
typedef struct _TSuperStateWithIDList
{
long NItems;
/* [size_is] */ TSuperStateWithID *pItem;
}TSuperStateWithIDList;
typedef struct TSuperStateWithID
{
TStateKeyList SuperState;
TDayOfWeekMask Days;
TTimeFilterList TimeFilters;
TDurationFilterList DurationFilters;
long ID;
}TSuperStateWithID;
typedef struct TStateKeyList
{
long NKeys;
/* [unique][size_is] */ TStateKey *pSK;
}TStateKeyList;
typedef long TDayOfWeekMask;
const long DOWSunday = 1;
const long DOWMonday = 2;
const long DOWTuesday = 4;
const long DOWWednesday = 8;
const long DOWThursday = 16;
const long DOWFriday = 32;
const long DOWSaturday = 64;
const TDayOfWeekMask DOWAllDays = DOWSunday + DOWMonday +
  DOWTuesday + DOWWednesday +
  DOWThursday + DOWFriday +
  DOWSaturday;
typedef struct _TTimeFilterList
{
TTimeZoneKey TZKey;
long NFilters;
[size_is(NFilters)] TTimeFilter* pFilters;
} TTimeFilterList;
typedef struct _TTimeFilter
{
TTimeOfDay Start;
TTimeOfDay Stop;
} TTimeFilter;
typedef long TTimeOfDay;// Minutes since midnight
typedef struct _TDurationFilterList
{
long NFilters;
[size_is(NFilters)] TDurationFilter* pFilters;
} TDurationFilterList;
typedef struct _TDurationFilter
{
TDuration Min;
TDuration Max;
} TDurationFilter;
typedef long TDuration;
typedef struct _TSegmentSelectorList
{
long NSelectors;
/* [unique][size_is] */ TSegmentSelector *pSegmentSelector;
}TSegmentSelectorList;
```

```
typedef struct_TSegmentSelector
{
BOOL SelectAllAssignedSchedules;
BOOL SelectAllUnassignedSchedules;
TEmployeeKeyList Employees;
TEmployeeGroupKeyList EmployeeGroups;
TStaffGroupKeyList StaffGroups;
TTrialScheduleKeyList TrialSchedules;
TTrialScheduleSetKeyList TrialScheduleSets;
BOOL UseOfficialSchedules;
}TSegmentSelector;
Note:See TSegmentSelector definition below.
typedef struct_TAbsoluteNomDateRangeList
{
long NRanges;
/* [unique][size_is] */ TAbsoluteNomDateRange *pNomDateRange;
}TAbsoluteNomDateRangeList;
typedef struct _TAbsoluteNomDateRange
{
TNomDate Start;
TNomDate Stop;
}TAbsoluteNomDateRange;
```

GetMomentBasedScheduleListTallyProvider Method

```
HRESULT GetMomentBasedScheduleListTallyProvider ( [in] PCTCSComString
pRequestor, [in] TMomentBasedTallyRequest* pRequest, [out]
ITCSMomentBasedScheduleListTallyProvider** ppProvider, [out]
TJobHandle* pJobHandle );
```

The GetMomentBasedScheduleTallyProvider method gets a moment-based schedule list tally provider for the moment-based tally request.

A related interface, ITCSMomentBasedScheduleListTallyProvider Interface, gets the schedule information given the pointer to the resolution provider.

Parameters

The following table describes the parameters of the GetMomentBasedScheduleListTallyProvider method.

Table 24: GetMomentBasedScheduleListTallyProvider Parameters

Parameter	Description
PCTCSComString pRequestor	String to identify source of the request
TMomentBasedTallyRequest* pRequest	Pointer to the request
ITCSMomentBasedScheduleListTallyProvider** ppProvider	Pointer to the Tally provider
TJobHandle* pJobHandle	Job Handle for debugging

Structure Definitions

The following are the structure definitions for this method. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct _TMomentBasedTallyRequest
      {
      TMomentBasedScheduleResolutionDefinition Definition;
```

```
TSuperStateWithIDList SuperStates;
PTCSComString pAlias;
PTCSComString pDescription;
}TMomentBasedTallyRequest;
typedef struct_TMomentBasedScheduleResolutionDefinition
{
TSegmentSelectorList SegmentSelectors;
TAbsoluteMomentRangeList AbsoluteMomentRanges;
TRelativeMomentRangeList RelativeMomentRanges;
BOOL MergeAssignedSchedules;
}TMomentBasedScheduleResolutionDefinition;
typedef struct TSuperStateWithIDList
{
long NItems;
/* [size_is] */ TSuperStateWithID *pItem;
}TSuperStateWithIDList;
typedef struct TSuperStateWithID
{
TStateKeyList SuperState;
TDayOfWeekMask Days;
TTimeFilterList TimeFilters;
TDurationFilterList DurationFilters;
long ID;
}TSuperStateWithID;
typedef struct TStateKeyList
{
long NKeys;
/* [unique][size_is] */ TStateKey *pSK;
}TStateKeyList;
typedef long TDayOfWeekMask;
const long DOWSunday = 1;
const long DOWMonday = 2;
const long DOWTuesday = 4;
const long DOWWednesday = 8;
const long DOWThursday = 16;
const long DOWFriday = 32;
const long DOWSaturday = 64;
const TDayOfWeekMask DOWAllDays = DOWSunday + DOWMonday +
  DOWTuesday + DOWWednesday +
  DOWThursday + DOWFriday +
  DOWSaturday;
typedef struct _TTimeFilterList
{
TTimeZoneKey TZKey;
long NFilters;
[size_is(NFilters)] TTimeFilter* pFilters;
} TTimeFilterList;
typedef struct _TTimeFilter
{
TTimeOfDay Start;
TTimeOfDay Stop;
} TTimeFilter;
typedef long TTimeOfDay;// Minutes since midnight
typedef struct _TDurationFilterList
{
long NFilters;
[size_is(NFilters)] TDurationFilter* pFilters;
} TDurationFilterList;
typedef struct _TDurationFilter
{
TDuration Min;
TDuration Max;
} TDurationFilter;
```

```
typedef long TDuration;
typedef struct_TSegmentSelectorList
{
long NSelectors;
/* [unique][size_is] */ TSegmentSelector *pSegmentSelector;
}TSegmentSelectorList;
typedef struct_TSegmentSelector
{
BOOL SelectAllAssignedSchedules;
BOOL SelectAllUnassignedSchedules;
TEmployeeKeyList Employees;
TEmployeeGroupKeyList EmployeeGroups;
TStaffGroupKeyList StaffGroups;
TTrialScheduleKeyList TrialSchedules;
TTrialScheduleSetKeyList TrialScheduleSets;
BOOL UseOfficialSchedules;
}TSegmentSelector;
Note:See TSegmentSelector definition below.
typedef struct_TAbsoluteMomentRangeList
{
long NRanges;
/* [unique][size_is] */ TAbsoluteMomentRange *pMomentRange;
}TAbsoluteMomentRangeList;
typedef struct TAbsoluteMomentRange
{
TMoment Start;
TMoment Stop;
}TAbsoluteMomentRange;
typedef struct TRelativeMomentRangeList
{
long NRanges;
[size_is ( NRanges ), unique] TRelativeMomentRange*
pMomentRange;
} TRelativeMomentRangeList;
typedef struct _TRelativeMomentRange
{
TDuration Start;
TDuration Stop;
} TRelativeMomentRange;
```

ITCSMomentBasedResolutionProvider Interface

The ITCSMomentBasedResolutionProvider interface resolves the schedule information requested in the GetMomentBasedResolutionProvider call in the perspective(s) requested in the GetMomentBasedResolutionProvider call. This interface is used in the Intra-Day Time Line module of the Aspect Workforce client.

To obtain an instance of the ITCSMomentBasedResolutionProvider interface, call the GetMomentBasedResolutionProvider method on the ITCSTallyServer interface.

GetResolvedScheduleInfo Method

```
HRESULT GetResolvedScheduleInfo ( [in, out] TUpdaterSequenceNumber*
pUpdaterSequenceNumber, [out] TFilteredScheduleResolutionList**
ppResolution )
```

The GetResolvedScheduleInfo method gets the resolved schedule information generated by GetMomentBasedResolutionProvider and using ITCSMomentBasedResolutionProvider pointer.

Parameters

The following table describes the parameters of the GetResolvedScheduleInfo method.

Table 25: GetResolvedScheduleInfo Parameters

Parameter	Description
TUpdaterSequenceNumber* pUpdaterSequenceNumber	Sequence number of the request
TFilteredScheduleResolutionList** ppResolution)	Pointer to the resolution list

Structure Definitions

The following are the structure definitions for this method. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct TFilteredScheduleResolutionList
        {
        long NSchedules;
        /* [size_is] */ TFilteredScheduleResolution
        *pFilteredScheduleResolution;
        }TFilteredScheduleResolutionList;
        typedef struct_TFilteredScheduleResolution
        {
        TResolutionIdentifier ResolutionIdentifier;
        long NPerspectives;
        /* [size_is] */ TTimePeriodResolutionForPerspectiveList
        *pTimePeriodResolutionForPerspectiveList;
        }TFilteredScheduleResolution;
        typedef struct TResolutionIdentifier
        {
        TEmployeeKey EmployeeSK;
        TTrialScheduleKey TrialScheduleSK;
        TTrialScheduleSetKey TrialScheduleSetSK;
        }TResolutionIdentifier;
        typedef struct_TTimePeriodResolutionForPerspectiveList
        {
        TPerspectiveKey PerspectiveSK;
        TStateKey DefaultStateSK;
        long NTimePeriods;
        /* [size_is] */ TTimePeriodResolutionForPerspective
        *pTimePeriodResolutionForPerspective;
        }TTimePeriodResolutionForPerspectiveList;
        typedef struct_TTimePeriodResolutionForPerspective
        {
        TTimePeriod TimePeriod;
        TStateKey StateSK;
        TSegmentCodeKey SegmentCodeSK;
        }TTimePeriodResolutionForPerspective;
        typedef struct_TTimePeriod
        {
        TAbsoluteMomentRange MomentRange;
        TNomDate NomDate;
        }TTimePeriod;
        typedef struct_TAbsoluteNomDateRange
        {
        TNomDate Start;
        TNomDate Stop;
        }TAbsoluteNomDateRange;
```

Note: as of version 22.0, TTimePeriodResolutionForPerspective also includes a TSegmentCodeKey value, SegmentCodeSK. This is the surrogate key of the segment code that was responsible for the resolved state. If you created an application that uses the ITCSMomentBasedScheduleResolutionProvider interface in a version prior to 22.0, you will need to update the application to be compatible with versions 22.0 and higher.

ITCSMomentBasedTimePeriodTallyProvider Interface

The ITCSMomentBasedTimePeriodTallyProvider interface tallies time spent in the state(s) requested in the GetMomentBasedTimePeriodTallyProvider call for the schedule information requested in the GetMomentBasedTimePeriodProvider call. This interface is used in the Intra-Day Performance module of the Aspect Workforce client.

To obtain an instance of the ITCSMomentBasedTimePeriodTallyProvider interface, call the GetMomentBasedTimePeriodTallyProvider method on the ITCSTallyServer interface.

GetTallyInfo Method

```
HRESULT GetTallyInfo ( [in, out] TUpdaterSequenceNumber*
pUpdaterSequenceNumber, [out] TMomentBasedTimePeriodTallyList**
ppTally )
```

The GetTallyInfo method gets the tally information for GetMomentBasedTimePeriodTallyProvider using the ITCSMomentBasedTimePeriodTallyProvider pointer.

Parameters

The following table describes the parameters of the GetTallyInfo method.

Table 26: GetTallyInfo Parameters

Parameter	Description
TUpdaterSequenceNumber* pUpdaterSequenceNumber	Sequence number of the resolution
TMomentBasedTimePeriodTallyList ** ppTally	Pointer to the tally information

Structure Definitions

The following are the structure definitions for this method. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct TMomentBasedTimePeriodTallyList
        {
        long NTallies;
        /* [size_is] */ TMomentBasedTimePeriodTally *pTimePeriodTally;
        }TMomentBasedTimePeriodTallyList;
        typedef struct TMomentBasedTimePeriodTally
        {
        long ID;
        TMomentBasedTimePeriodHeadCountList TimePeriodHeadCountList;
        }TMomentBasedTimePeriodTally;
        typedef struct_TMomentBasedTimePeriodHeadCountList
        {
        long NTimePeriods;
        /* [size_is] */ TMomentBasedTimePeriodHeadCount
        *pTimePeriodHeadCount;
```

```
}TMomentBasedTimePeriodHeadCountList;
typedef struct_TMomentBasedTimePeriodHeadCount
{
TAbsoluteMomentRange MomentRange;
long HeadCount;
}TMomentBasedTimePeriodHeadCount;
typedef struct_TAbsoluteNomDateRange
{
TNomDate Start;
TNomDate Stop;
}TAbsoluteNomDateRange;
```

ITCSNomDateBasedScheduleListTallyProvider Interface

The ITCSNomDateBasedScheduleListTallyProvider interface tallies time spent in the state(s) requested in the GetNomDateBasedScheduleListTallyProvider call for the schedule information requested in the GetNomDateBasedScheduleListTallyProvider call. This interface is used in Agent Productivity reports in the Aspect Workforce client.

To obtain an instance of the ITCSNomDateBasedScheduleListTallyProvider interface, call the GetNomDateBasedScheduleListTallyProvider method on the ITCSTallyServer interface.

GetTallyInfo Method

```
HRESULT GetTallyInfo ( [in, out] TUpdaterSequenceNumber*
pUpdaterSequenceNumber, [out] TScheduleSuperStateTimePeriodsList**
ppTally )
```

The GetTallyInfo method gets the tally information for GetNomDateBasedScheduleTallyProvider using the ITCSNomDateBasedScheduleListTallyProvider pointer.

Parameters

The following table describes the parameters of the GetTallyInfo method.

Table 27: GetTallyInfo Parameters

Parameter	Description
TUpdaterSequenceNumber* pUpdaterSequenceNumber	Sequence number of the resolution
TScheduleSuperStateTimePeriodsList ** ppTally	Pointer to the tally information

Structure Definitions

The following are the structure definitions for this method. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct_TScheduleSuperStateTimePeriodsList
	{
	long NSchedules;
	/* [size_is] */ TScheduleSuperStateTimePeriods
	*pScheduleSuperStateTimePeriods;
	}TScheduleSuperStateTimePeriodsList;
	typedef struct_TScheduleSuperStateTimePeriods
	{
	TResolutionIdentifier ResolutionIdentifier;
	TSuperStateTimePeriodsList SuperStateTimePeriods;
	}TScheduleSuperStateTimePeriods;
```

```
typedef struct_TSuperStateTimePeriodsList
{
long NSuperStates;
/* [size_is] */ TSuperStateTimePeriods *pSuperStateTimePeriods;
}TSuperStateTimePeriodsList;
typedef struct_TSuperStateTimePeriods
{
long ID;
TTimePeriodList TimePeriods;
}TSuperStateTimePeriods;
typedef struct_TTimePeriodList
{
long NTimePeriods;
/* [size_is] */ TTimePeriod *pTimePeriod;
}TTimePeriodList;
```

ITCSNomDateBasedScheduleTallyProvider Interface

The ITCSNomDateBasedSchedulesTallyProvider interface calculates daily totals of the time spent in the state(s) requested in the GetNomDateBasedScheduleTallyProvider call for the schedule information requested in the GetNomDateBasedScheduleTallyProvider call. This interface is used to calculate superstate hours.

To obtain an instance of the ITCSNomDateBasedScheduleTallyProvider interface, call the GetNomDateBasedScheduleTallyProvider method on the ITCSTallyServer interface.

GetTallyInfo Method

```
HRESULT GetTallyInfo ( [in, out] TUpdaterSequenceNumber*
pUpdaterSequenceNumber,[out] TNomDateBasedScheduleTallyList** ppTally
);
```

The GetTallyInfo method gets the tally information for GetNomDateBasedScheduleTallyProvider method using the ITCSNomDateBasedScheduleTallyProvider pointer

Parameters

The following table describes the parameters of the GetTallyInfo method.

Table 28: GetTallyInfo Parameters

Parameter	Description
TUpdaterSequenceNumber* pUpdaterSequenceNumber	Sequence number of the resolution
TNomDateBasedScheduleTallyList** ppTally	Pointer to the tally information

Structure Definitions

The following are the structure definitions for this method. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct_TNomDateBasedScheduleTallyList
        {
        long NSchedules;
        /* [size_is] */ TNomDateBasedScheduleTally *pScheduleTally;
        }TNomDateBasedScheduleTallyList;
        typedef struct_TNomDateBasedScheduleTally
        {
```

```
TResolutionIdentifier ResolutionIdentifier;
TNomDateBasedScheduleTallyForSuperStateList
ScheduleTallyForSuperStates;
}TNomDateBasedScheduleTally;
typedef struct _TNomDateBasedScheduleTallyForSuperStateList
{
long NSuperStates;
/* [size_is] */ TNomDateBasedScheduleTallyForSuperState
*pScheduleTallyForSuperState;
}TNomDateBasedScheduleTallyForSuperStateList;
typedef struct _TNomDateBasedScheduleTallyForSuperState
{
long ID;
TNomDateBasedTimeTotalList TimeTotals;
}TNomDateBasedScheduleTallyForSuperState;
typedef struct_TNomDateBasedTimeTotalList
{
long NNomDates;
/* [size_is] */ TNomDateBasedTimeTotal *pTimeTotal;
}TNomDateBasedTimeTotalList;
typedef struct_TNomDateBasedTimeTotal
{
TNomDate NomDate;
long NumberOfMinutes;
}TNomDateBasedTimeTotal;
```

ITCSMomentBasedScheduleListTallyProvider Interface

The ITCSMomentBasedScheduleListTallyProvider interface tallies time spent in the state(s) requested in the GetMomentBasedScheduleListTallyProvider call for the schedule information requested in the GetMomentBasedScheduleListTallyProvider call.

To obtain an instance of the ITCSMomentBasedScheduleListTallyProvider interface, call the GetMomentBasedScheduleListTallyProvider method on the ITCSTallyServer interface.

GetTallyInfo Method

```
HRESULT GetTallyInfo ( [in, out] TUpdaterSequenceNumber*
pUpdaterSequenceNumber, [out] TScheduleSuperStateTimePeriodsList**
ppTally )
```

The GetTallyInfo method gets the tally information for GetMomentBasedScheduleTallyProvider using the ITCSMomentBasedScheduleListTallyProvider pointer.

Parameters

The following table describes the parameters of the GetTallyInfo method.

Table 29: GetTallyInfo Parameters

Parameter	Description
TUpdaterSequenceNumber* pUpdaterSequenceNumber	Sequence number of the resolution
TScheduleSuperStateTimePeriodsList ** ppTally	Pointer to the tally information

Structure Definitions

The following are the structure definitions for this method. To obtain the results, the members of the appropriate structure need to be populated.

```
typedef struct TScheduleSuperStateTimePeriodsList
	{
	long NSchedules;
	/* [size_is] */ TScheduleSuperStateTimePeriods
	*pScheduleSuperStateTimePeriods;
	}TScheduleSuperStateTimePeriodsList;
	typedef struct TScheduleSuperStateTimePeriods
	{
	TResolutionIdentifier ResolutionIdentifier;
	TSuperStateTimePeriodsList SuperStateTimePeriods;
	}TScheduleSuperStateTimePeriods;
	typedef struct_TSuperStateTimePeriodsList
	{
	long NSuperStates;
	/* [size_is] */ TSuperStateTimePeriods *pSuperStateTimePeriods;
	}TSuperStateTimePeriodsList;
	typedef struct_TSuperStateTimePeriods
	{
	long ID;
	TTimePeriodList TimePeriods;
	}TSuperStateTimePeriods;
	typedef struct TTimePeriodList
	{
	long NTimePeriods;
	/* [size_is] */ TTimePeriod *pTimePeriod;
	}TTimePeriodList;
```

Common Type and Property Definitions

This section contains structure definitions for common types and data types for common properties.

Common Type Structure Definitions

```
typedef struct _TSegmentSelector
	{
	BOOL SelectAllAssignedSchedules;
	BOOL SelectAllUnassignedSchedules;
	TEmployeeKeyList Employees;
	TEmployeeGroupKeyList EmployeeGroups;
	TStaffGroupKeyList StaffGroups;
	TTrialScheduleKeyList TrialSchedules;
	TTrialScheduleSetKeyList TrialScheduleSets;
	BOOL UseOfficialSchedules;
	}TSegmentSelector;
	typedef struct_TEmployeeKeyList
	{
	long NKeys;
	/* [unique][size_is] */ TEmployeeKey *pSK;
	}TEmployeeKeyList;
	typedef struct_TEmployeeGroupKeyList
	{
	long NKeys;
	/* [unique][size_is] */ TEmployeeGroupKey *pSK;
	}TEmployeeGroupKeyList;
	typedef struct_TStaffGroupKeyList
	{
	long NKeys;
	/* [unique][size_is] */ TStaffGroupKey *pSK;
```

```
}TStaffGroupKeyList;
typedef struct_TTrialScheduleKeyList
{
long NKeys;
/* [unique][size_is] */ TTrialScheduleKey *pSK;
}TTrialScheduleKeyList;
typedef struct_TTrialScheduleSetKeyList
{
long NKeys;
/* [unique][size_is] */ TTrialScheduleSetKey *pSK;
}TTrialScheduleSetKeyList;
```

Data Types of Common Properties

The following table shows the data types for common properties.

Table 30: Common Property Data Types

Property Name	Data Type
TDayCount	Long
TDuration	Long
TTimeOfDay	Long, 0=Midnight to 1439=11:59 PM
TMoment	Long
TNomDate	Long
TDayOfWeekMask	Long, bitmask, a value of 2^{DOW} means that DOW should be included. DOW values: 0=Sunday, 1=Monday, etc. A value of 127 should be supplied if the superstate applies on all days of the week.
TTCSKey	Double
TUpdaterSequenceNumber	TTCSKey
TEmployeeKey	TTCSKey
TEmployeeGroupKey	TTCSKey
TEmployeeGroupTreeKey	TTCSKey
TPerspectiveKey	TTCSKey
TScheduleResolutionDefinitionKey	TTCSKey
TSegmentKey	TTCSKey
TSegmentCodeKey	TTCSKey
TSegmentSelectorKey	TTCSKey
TStaffGroupKey	TTCSKey
TStateKey	TTCSKey

Property Name	Data Type
TTallyDefinitionKey	TTCSKey
TTrialScheduleKey	TTCSKey
TTrialScheduleSetKey	TTCSKey
TTimeZoneKey	TTCSKey
TDetailSegmentKey	TSegmentKey
TGeneralSegmentKey	TSegmentKey

Sample Code and Required Files

This section contains sample code for the Tally Server API and required files.

Sample Code

This section contains sample code for the Tally Server API.//Initialize COM

```
OleInitialize(NULL);//Create an instance of tally server

hr = CoCreateInstance ( CLSID_TCSTallyServer,
       NULL,
       CLSCTX_ALL,
       IID_ITCSTallyServer,
       (void**) &m_pTallyServer );

if ( FAILED ( hr ) )

{
       CString Msg;
       Msg.Format("TalleyServer CoInitializeEx failure %x", hr);
       AfxMessageBox(Msg);

}

//Create a segment selector

SegmentSelector::CSegmentSelector ss;

ss.UseOfficialSchedules ( true );//Surrogate Key for the Desired
Staff Group

ss.AddStaffGroup  (-102147482579.0);// Crteate a
MomentBasedScheduleResolutionRequest

ScheduleResolutionDefinition::
CMomentBasedScheduleResolutionRequest mbsrr;

mbsrr.AddSegmentSelector ( ss );// Create nominal dates based on
desired date/time combination

// STARTDATE 00:00:00 to STOPDATE 23:59:59

StringToCTime("01/01/2001", "00:00:00", m_StartTime);

StringToCTime("01/31/2001", "23:59:59", m_StopTime);

long StartMoment = ConvertCTimeToMoment(m_StartTime);

long StopMoment = ConvertCTimeToMoment(m_StopTime);
```

```
mbsrr.AddAbsoluteMomentRange ( StartMoment, StopMoment );

mbsrr.SetDatabaseAlias ( std::wstring(L"S5DATA"));

mbsrr.SetDescription ( std::wstring ( L"IDP schedule resolution for
RTA" ));//Surrogate Key for the Desired Perspective

mbsrr.AddPerspective( -102147483315.0);//Create the
TmomentBasedScheduleResolutionRequest

TMomentBasedScheduleResolutionRequest Request=mbsrr;

CComPtr<ITCSMomentBasedResolutionProvider> pProvider;

TJobHandle JobHandle;// Submit the request to tally server.

hr = m_pTallyServer->GetMomentBasedResolutionProvider ( L"RTA" ,
&Request, &pProvider, &JobHandle );//Release the request when done.

ScheduleResolutionDefinition::
ReleaseMomentBasedScheduleResolutionRequest(Request);

if (SUCCEEDED(hr))

{

//Declare a schedule pointer and the sequence number
       TFilteredScheduleResolutionList* pResSched=NULL;
       TUpdaterSequenceNumber SeqNum =

TCSUtil::TUpdaterSequenceNumberMin;//Get the resolved schedule
information
       hr = pProvider->GetResolvedScheduleInfo(&SeqNum, &pResSched);
       if (SUCCEEDED(hr))
       {

//Iterate through the schedule information

for (int i=0; i<pResSched->NSchedules; ++i)
       {

TFilteredScheduleResolution* EmployeeSchedules = &pResSched->
pFilteredScheduleResolution[i];
       }
       }

}

m_pTallyServer.Release();
```

Required Files

The files needed to build a tally server client and attach to the Tally Server include:

- CLAS_DOWFilter.h
- CLAS_DOWFilter_fwd.h
- CLAS_DurationFilter.cpp
- CLAS_DurationFilter.h
- CLAS_DurationFilter_fwd.h
- CLAS_KeyLists.h
- CLAS_Moments.cpp
- CLAS_Moments.h
- CLAS_Moments_fwd.h

- CLAS_NomDate.cpp
- CLAS_NomDate.h
- CLAS_NomDate_fwd.h
- CLAS_ScheduleResolutionDefinition.cpp
- CLAS_ScheduleResolutionDefinition.h
- CLAS_ScheduleResolutionDefinition_fwd.h
- CLAS_SegmentSelector.cpp
- CLAS_SegmentSelector.h
- CLAS_SegmentSelector_fwd.h
- CLAS_SuperStateWithID.cpp
- CLAS_SuperStateWithID.h
- CLAS_SuperStateWithID_fwd.h
- CLAS_TallyDefinition.cpp
- CLAS_TallyDefinition.h
- CLAS_TallyDefinition_fwd.h
- CLAS_TallyExceptions.h
- CLAS_TallyTypeReleases.cpp
- CLAS_TallyTypeReleases.h
- CLAS_TCSConstants.h
- CLAS_TimeFilter.cpp
- CLAS_TimeFilter.h
- CLAS_TimeFilter_fwd.h
- StringFunctions.cpp
- StringFunctions.h
- TallyServerSampleCPP.cpp
- TallyServerSampleCPP.vcxproj
- TallyTransport.h
- TallyTransport.idl
- TallyTransport_i.c
- TallyTransport_p.c
- TallyTypes.h
- TallyTypes.idl
- TCSJobCommon.h
- TCSJobCommon.idl

The files are from a Visual Studio 2019 project. Since the GUIDs used by the 32 bit Tally Transport interfaces differ from the 64 bit Tally Transport, you will need to inspect the IDL files provided with this

SDK to see the values of the GUIDs needed. The 32 bit GUIDs are unchanged from earlier versions of WFM; the 64 bit GUIDs are new to reflect the change in the data packing from 4 byte packing to 8 byte packing made for 64 bit in WFM 8.2 and later.

Updater API Developer Guide

This guide specifies the Updater API for development of third-party applications with Aspect Workforce. The Updater API enables you to add, make changes to, and delete segments that comprise agents' schedules.

While the discussion below provides sufficient details to use the Updater API to submit schedule changes, we have deprecated the use of these interfaces by outside software. We, instead, strongly recommend that you consider using the ScheduleSegmentFilter and ScheduleSegmentUpdate functions from the Alvaria Web Services. Alvaria Web Services are described in chapters **0** and **0**. You will find information on specific web service functions in the help file, WFM220_WebServices_Help.chm, delivered with this SDK. The ScheduleSegmentsFilter and SegmentSegmentUpdate functions are described in the help file under OfficialSchedules Server. We include the following discussions of the Updater API to aid in your understanding and use of the related functions in the Alvaria Web Services.

Related Software Files

We no longer deliver source files to aid your use of the Updater API. We strongly recommend that you use the functions in the Alvaria Web Services discussed above.

Prerequisites

The following table shows the requirements for using the Updater API.

Table 1: Aspect Workforce Requirements for the Updater API

Versions	Aspect Workforce 7.5 or later
Enhancement Packages	None
Database	Aspect Workforce database must be installed
Other Software	• Updater service must be installed • Aspect Workforce client must be installed

Additional Requirements

In addition to the prerequisites, the Updater API has the following requirements relating to hardware, software, connectivity, and security:

- A computer that meets the requirements for the Aspect Workforce client software.
- The Aspect Workforce client installed on the computer from where you will use the Updater API.
- While the Updater API does not run the Aspect Workforce client (TCS.EXE), it does require several files included in the Aspect Workforce installation.

- Network connections to the Aspect Workforce database server and application server(s).
- An Aspect Workforce username and password for use with the Updater API. If you do not have an Aspect Workforce account, you must ask your administrator to create one for you.
- Database software drivers that allow your software tools to communicate with your Aspect Workforce database.

Audience

The Updater API can be used by third-party developers and others to add, make changes to, and delete segments that comprise agents' schedules in the Aspect Workforce system. The Updater API is used by the Aspect Workforce client to make changes to agents' schedules using the Official Schedule Editor, the Segment Worksheet, and the Trial Schedule Manager. The same API, documented here, can be used by application developers at a customer site to make changes to agents' schedules while still utilizing the rules-checking engine hosted in the Updater server. The Updater server is one of several services that comprise the Aspect Workforce application server.

This document assumes that you are familiar with the following topics:

- Basic database concepts
- SQL programming
- COM concepts and programming methods

For more information on these topics, please consult the documentation provided with the respective products or the large number of books available on the topics.

Additional Documentation

The documentation for the Updater API consists of this developer guide. However, general Aspect Workforce concepts are explained in the Aspect Workforce User Guide, and the Aspect Workforce online Help.

Overview

The Updater API is designed to provide a stable and easy-to-use environment in which changes, additions, and deletions to agents' schedules can be made. The Updater server and client together ensure that the various rules regarding segments defined by the Aspect Workforce administrator are always enforced. This ensures that the segments are consistently in line with customers' policies. The Updater API also allows for the integration of third-party developed applications with the Aspect Workforce system. These might include applications that import schedule data from time-keeping systems, automated attendance tracking systems, etc.

Before using the Updater API, you must be familiar with the basic concepts of schedules and segments in Aspect Workforce. This document outlines some of concepts that are fundamental in understanding schedules and the way that they are stored and manipulated in Aspect Workforce.

Components of the Updater API

The Updater API consists of this document and the COM Dlls installed with Aspect Workforce that implement the interfaces described in this chapter.

Installing the Updater API

Installing the Updater API is a straightforward process. Perform the following steps on the computer where you want to access the Updater API:

1. Install the Aspect Workforce client software. This is done by running the Setup program from the TCSClient share on your application server.

2. You may need to copy the following type libraries from the Aspect Workforce system: UpdaterClient.tlb and UpdaterCommon.tlb. The default location of these files is “C:\Program Files\Common Files\Alvaria Shared”.

3. Install the database software driver needed to communicate with your Aspect Workforce database.

4. Perform any configuration needed so that your software can connect to your Aspect Workforce database.

Concepts in Aspect Workforce

There are several fundamental Aspect Workforce concepts that you must understand before you can successfully use the Updater API. Some of these concepts are also explained in the Aspect Workforce online Help.

Surrogate Keys (SK)

The primary key in most of the Aspect Workforce tables is a surrogate key (SK). Surrogate keys are defined as the NUMERIC data type and must be obtained from either the GET_SK or the GET_SKS stored procedures. The extensive use of surrogate keys in the Aspect Workforce data model permits the easy changing of properties that Aspect Workforce users normally identify as unique, which are stored instead as alternate keys in the database. For example, the primary key in the employee table (EMP) is a generated surrogate key whereas the employee id field (ID), supplied by the user, is an alternate key. It is the primary key of the employee table (EMP_SK), which appears in composite alternate and composite foreign keys in other tables. Hence, the user can safely modify the employee’s ID without jeopardizing the integrity of the data in any of the other tables. Surrogate keys are not revealed to the user in the Aspect Workforce client software and are of use only to someone who needs to access or modify Aspect Workforce data directly or through the Updater API.

Schedules and Segments

A schedule is a record of what an employee is supposed to do on any given day and is simply a collection of segments for that date. Segments are the basic building blocks of schedules. They are the essential records you create to track how employees are spending time and how well you are achieving your staffing objectives. To record the fact that an employee is attending a meeting, for example, you would enter a segment for that employee that includes the date, time period, and the appropriate segment code (defined later).

UTC Time

All times in Aspect Workforce are stored as some representation of Universal Coordinated Time (UTC), which was previously referred to as Greenwich Mean Time or GMT. By exclusively storing UTC times in the database it is possible to store the start and stop times for segments assigned to employees working in several different time zones in the same table, without regard to those time zones. When a user enters

a time in the Aspect Workforce client it is converted to UTC using the local time zone setting of the user before being stored in the database. These times are then converted back to the local time for display to the user.

Two units of time stored in the Aspect Workforce database relate to segments and schedules: Moments and Nominal Dates.

Moments

All start and stop times for segments in Aspect Workforce are stored as “moments”. A moment is defined to be the UTC time represented as the number of minutes since 12:00 AM December 30 1899. Therefore, the unit of moments is exactly one minute.

Nominal Dates

The nominal date of a segment is the date that a segment is associated with for reporting and selection purposes. Nominal dates are represented as the number of days since December 30, 1899. Therefore, the unit of nominal dates is exactly one day.

Segment Codes

Segment Codes are used to identify the type of activity that a segment represents. For example the segment code “UNPBRK” may be used to represent unpaid breaks. Each segment must be associated with a segment code.

Segment Categories

Segment categories are broad characterizations of segment codes that enable you to define generalized segment entry rules. They allow the definition of rules at a higher level than possible with just segment codes. For example, to set up a rule that forces segments of category “break” to be contained in segments of the category “work” you can associate all the break segment codes with the break category and all the work segment codes with the work category and them set up just one rule.

Segment Entry Rules

Segment entry rules are rules that you define which Updater enforces whenever a change or an addition is made to an employee’s official segment data. There are two categories of segment entry rules: containment rules and overlap rules. Segment containment rules allow you to define rules that force segments of a particular category to be completely contained in segments of another category. Segment overlap rules allow you to define rules that do not allow segments of a particular category to overlap segments of another category.

For example, you could define a containment segment entry rule that forced all segments that are of the category “TRNG” to be completely contained in segments that are of the category “WORK”. This would define a rule that would force all training related segments to be contained in work segments. Similarly, you could define an overlap segment entry rule that prohibited segments of category “TRNG” to overlap segment of category “VACA”, and so on.

Segment Types

There are two types of segments: General segments and Detail segments. Any segment code can be used to enter either a General or a Detail segment.

General Segments

A general segment is a segment that does not have a start or a stop moment but is associated with a particular nominal date. Any segment can be entered as a general segment, which is in effect whenever the employee has at least one detail segment for the same nominal date. General segments are used to denote “activity” that is valid for the entire day’s schedule for a person. For example, to denote that a person is on vacation for the day regardless any other activity previously assigned to the person, you could enter a General segment with a vacation code.

Detail Segments

A detail segment is a segment that is not only associated with a nominal date but also has a start and a stop moment. The start and stop moments delineate the absolute time that the segment is in effect.

Data Types and Stored Procedures

This section discusses data types and stored procedures.

About Data Types

Aspect Workforce defines and uses several data types throughout its schema. Some of these data types are defined below. Note that these data types may be defined slightly differently on the different databases that Aspect Workforce supports. Also note that the names of all objects in some databases, such as Oracle, will be prefixed by the owner/creator of the schema, which is typically TCSDBOWNER.

Refer to the Aspect Workforce public data model for the most up-to-date information on all the tables in the Aspect Workforce schema. (For more information, see the *Aspect Workforce Public Data Model Guide,* available on the Alvaria Product Document Library.) The public data model also contains information about the primary, alternate, and foreign keys in each of the tables, and the relationships between tables.

Table 2: Data Types

Data Type	Domain	Description
SK	NUMERIC(15,1)	Surrogate Key: Used for the primary key of most tables. The SK is generated using either the GET_SK or the GET_SKS stored procedures to guarantee unique numbers.
NOM_DATE	NUMERIC(15,1)	Nominal Date: Used with both General and Detail segments to denote the day for which the schedule is in effect.
BOOLEAN	NCHAR(1)	Can be either ‘T’ or ‘F’, for true or false.

Data Type	Domain	Description
MEMO	BLOB SUB_TYPE 1 SEGMENT SIZE 80	Memo field: Used for entering memos for segments and other records. Columns with the data type of *Memo* are implemented as NCLOB in Oracle and nvarchar(max) in SQL Server.
START_MOMENT	NUMERIC(15,1)	Start Moment: Used to define the start moment for a segment.
STOP_MOMENT	NUMERIC(15,1)	Stop Moment: Used to define the stop moment for a segment.
TCS_MOMENT	NUMERIC(15,1)	Moment: Used to indicate a moment.
CODE	NVARCHAR(30)	Code: Used to indicate a character-based code. This is used as the data type for several fields, including segment code and segment category.
DESCRIPTION	NVARCHAR(30)	Description: Used for the descriptions of fields, usually those defined by type CODE.
TCS_POS_INTEGER	NUMERIC(15,1)	Positive integer: Used to define a field that must have a positive integer.
COLOR	NUMERIC(15,1)	Color: Used for the display of segments.
FONT_COLOR	NUMERIC(15,1)	Font Color: Used for the display of segments.

About Stored Procedures and Surrogate Keys

The Aspect Workforce data model defines several stored procedures. However, only the ones that you will need to use when using the Updater API are described in the following section.

Most of the primary keys in the Aspect Workforce database are surrogate keys. These keys must be unique in each table. There are two stored procedures that must be used to generate the surrogate keys used in Aspect Workforce.

Caution: Do not generate your own keys for use as primary keys as this may generate errors when you save your data.

GET_SK

The GET_SK procedure is used to generate a single surrogate key. It has one output parameter, SK_OUT (of type SK), which returns the generated SK.

GET_SKS

The GET_SKS procedure is used to generate a number of surrogate keys. This function is useful and faster if you need to generate a larger number of keys. It has one input parameter, NUM_SKS_IN, that specifies the number of keys to generate; it has one output parameter, SK_OUT, which is the first key in a continuous series of keys reserved for the process that called GET_SKS.

Database Tables

This section defines several database tables that are relevant to the Updater API.

Official Segment Tables

This section describes official segment tables that are relevant to the Updater API.

EMP

The EMP (Employee Records) table is an Official Segment table. This is the master table containing a list of all the employees in your Aspect Workforce system.

Table 3: EMP (Employee Records) Table

Column Name	Data Type	Description
EMP_SK	SK	Employee Surrogate Key
ID	NCHAR(10)	Employee ID (entered by user)
LAST_NAME	NVARCHAR(30)	Employee last name
FIRST_NAME	NVARCHAR(30)	Employee first name
EFF_HIRE_NOM_DATE	NOM_DATE	Employee effective-hire nominal date
TERM_NOM_DATE	NOM_DATE	Employee termination nominal date
ACTIVE_FLAG	BOOLEAN	Flag to indicate whether the employee is active or not
SHORT_NAME	NVARCHAR(30)	Abbreviated form of employee's name. Used by the Aspect Workforce client for display and in reports
SORT_NAME	NVARCHAR(60)	Alternate name for employee used by the Aspect Workforce client during sorting
TIME_ZONE_SK	SK	Surrogate key for time zone that the employee works in
SENIORITY	NVARCHAR(12)	Employee seniority
EMAIL_ADR	NVARCHAR(255)	Employee email address

Column Name	Data Type	Description
MEMO	MEMO	Additional, unstructured information stored for this employee
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record
UPD_TS	DATE	Date and time at which the associated record was updated

GEN_SEG

The GEN_SEG (General Segments) table is an Official Segments table. It contains all the official general segments in your Aspect Workforce system.

Table 4: GEN_SEG (General Segments) Table

Column Name	Data Type	Description
GEN_SEG_SK	SK	General Segment surrogate key
SEG_CODE_SK	SK	Segment Code surrogate key (segment code for general segment)
EMP_SK	SK	Employee surrogate key (employee assigned to general segment)
NOM_DATE	NOM_DATE	Nominal Date for general segment
MEMO	MEMO	Memo associated with general segment

DET_SEG

The DET_SEG (Detail Segments) table is an Official Segments table. It contains all the official detail segments in your Aspect Workforce system.

Table 5: DET_SEG (Detail Segments) Table

Column Name	Data Type	Description
DET_SEG_SK	SK	Detail Segment surrogate key
SEG_CODE_SK	SK	Segment Code surrogate key (segment code for detail segment)
EMP_SK	SK	Employee surrogate key (employee assigned to detail segment)
NOM_DATE	NOM_DATE	Nominal Date for detail segment
START_MOMENT	START_MOMENT	Start Moment for Detail Segment
STOP_MOMENT	STOP_MOMENT	Stop Moment for Detail Segment

Column Name	Data Type	Description
MEMO	MEMO	Memo associated with general segment

DLY_SEG_SMRY

The DLY_SEG_SMRY (Daily Segment Summary) table is an Official Segments table. It contains summaries of all the official segments on a per-employee per-nominal date basis.

Table 6: DLY_SEG_SMRY (Daily Segment Summary) Table

Column Name	Data Type	Description
NOM_DATE	NOMINAL_DATE	Nominal Date for Daily Segment summary
EMP_SK	SK	Employee for Daily Segment Summary
EARL_START_MOMENT	TCS_MOMENT	Earliest Start Moment for employee for Nominal Date
LATE_STOP_MOMENT	TCS_MOMENT	Latest Stop Moment for employee for Nominal Date
GEN_SEG_CODE_SK	SK	Highest ranking General Segment for Nominal Date
WORK_SEG_CODE_SK	SK	Highest ranking work segment

Segment Information Tables

This section describes segment information tables that are relevant to the Updater API.

SEG_CODE

The SEG_CODE (Segment Code) table is a Segment Information table. It contains the segment code definitions used in creating general and detail segments.

Table 7: SEG_CODE (Segment Code) Table

Column Name	Data Type	Description
SEG_CODE_SK	SK	Segment code surrogate key
CODE	CODE	Character representation of Segment Code. This is what is entered by users when creating segments
RANK	TCS_POS_INTEGER	The rank ordering of segment, used in resolving overlapping segments

Column Name	Data Type	Description
DESCR	DESCRIPTION	Brief description of segment code
REQ_MEMO	BOOLEAN	Determines whether a memo is required when entering a segment with this code
COLOR	COLOR	Background color when displaying the segment code
FONT_COLOR	FONT_COLOR	Font color for the segment code letters when displayed.
DEF_DUR	TCS_POS_INTEGER	Default duration (in minutes) of a segment. Must be between 1 and 1439.
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record.
UPD_TS	DATE	Date and time at which the associated record was updated.

SEG_CAT

The SEG_CAT (Segment Category) table is a Segment Information table. It contains the definitions of segment categories that can be used to define segment entry rules.

Table 8: SEG_CAT (Segment Category) Table

Column Name	Data Type	Description
SEG_CAT_SK	SK	Segment Category surrogate key
CODE	CODE	Segment Category code
DESCR	DESCRIPTION	Description if segment category
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record
UPD_TS	DATE	Date and time at which the associated record was updated

SEG_CODE_CAT_ASSOC

The SEG_CODE_CAT_ASSOC (Code/Category Association) table is a Segment Information table. It contains the associations between the segment categories and segment codes.

Table 9: SEG_CODE_CAT_ASSOC (Code/Category Association) Table

Column Name	Data Type	Description
SEG_CAT_SK	SK	Segment Category surrogate key

Column Name	Data Type	Description
SEG_CODE_SK	SK	Segment Code surrogate key

SEG_CAT_RULE

The SEG_CAT_RULE (Segment Category Rule) contains the rules that the Updater server and client enforce for every official segment that is entered in the Aspect Workforce system.

Table 10: SEG_CAT_RULE (Segment Category Rule) Table

Column Name	Data Type	Description
SEG_CAT_RULE_SK	SK	Segment Category Rule surrogate key
SEG_CAT_SK1	SK	Surrogate key for first Segment Category in rule
SEG_CAT_SK2	SK	Surrogate key for second Segment Category in rule
RULE_TYPE	ANSI_NCHAR_02	Type of rule. Must be either IN or OV[a]
SIGN	ANSI_NCHAR_01	How the rule is interpreted. Must be either + or -[b]
ENF_TYPE	ANSI_NCHAR_02	Enforcement type. Must be ST or WA. Specifies whether a violation of this rule will result in the segment not being saved or be saved with a warning message generated by the Updater server
UPD_TCS_LOGIN_SK	SK	Aspect Workforce user who updated the associated record
UPD_TS	DATE	Date and time at which the associated record was updated

[a] Of the four possible combinations of RULE_TYPE and SIGN, only two are currently allowed. They are:

RULE_TYPE: IN, SIGN: +: Defines a must be contained in rule. That is, segments of category SEG_CAT_SK1 must be contained in segments of category SEG_CAT_SK2.

RULE_TYPE: OV, SIGN: -: Meaning: Defines a must not overlap rule. That is, segments of SEG_CAT_SK1 must not overlap segments of category SEG_CAT_SK2.

b See footnote a.

Updater Architecture

The Updater server is designed in a multi-tier architecture. The Aspect Workforce clients and any other applications that want to make changes to employees' official segments do not write the data directly to the database. Instead, they communicate with the Updater server, which enforces the rules set up in the Aspect Workforce system to ensure that data integrity is maintained and the customer's segment entry rules are enforced. Clients of the Updater server, including the Aspect Workforce client, that want to make

changes to official segments, communicate with it using the supplied Updater Client. This is done to shield the client from the complexity of the Updater server's interface, as well as to perform certain validation on the client. This prevents the server from having to do processing that it does not need to perform. For example, there are certain rules that can be processed independent of the rules that customers create; these rules are enforced by the Updater client. The user-defined segment category rules are enforced by the Updater server, which also ensures that requestsare modified in the order they are received.

The following diagram illustrates the Updater architecture:

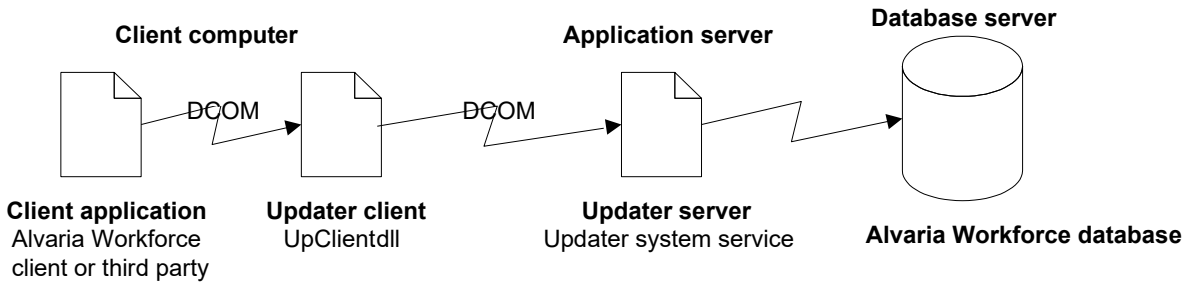


Figure 1: Updater Architecture

For Aspect Workforce 8.2 and later, The Updater service is running as a 64-bit system service. Both 32-bit and 64-bit clients are supported for this version.

Updater Server

The Updater server is a Windows system service that is designed to handle concurrent requests from many Updater clients. Client applications do not communicate with the Updater server directly. Rather, they communicate with the Updater client, which in turn communicates with the Updater server. The Updater server queues every transaction it receives and performs validation on them in the order they are received. When it is finished evaluating a transaction, and if the transaction does not violate any rules, it saves the changes to the database and sends a success message back to the Updater client. If any segment in the transaction violates any rules that are of strict enforcement type, Updater server fails the transaction, doesn't save any changes to the database, and returns a list of errors back to the Updater client. If the only violations the Updater server encounters while evaluating a transaction are of warning enforcement type then Updater server saves the changes and sends a list of the warnings back to the Updater client.

Updater Client

The Updater client and the Updater client type library are both contained in a single file called UpClient.dll. The Updater client type library is installed and registered when the Aspect Workforce client is installed on the computer. Client applications communicate with the Updater Client using a DCOM dual-interface. Third-party client applications can also use the wrappers provided. There are two versions of UpdaterClient.dll installed, one for 64-bit clients and one for 32-bit clients. The 64-bit version is installed in C:\Program Files\Common Files\Alvaria Shared, and the 32-bit version is installed in C:\Program Files (x86)\Common Files\Alvaria Shared.

When you add segment changes, additions, or deletions to a transaction, the Updater client holds the transaction locally until the transaction is submitted. If no rules are violated in local, client-side processing it then packages and sends the transaction request to the Updater server. The server performs additional rule checking on the transaction before either failing the transaction or committing it to the database. If

there are any rule violations the Updater client reports the warnings or errors back to the client application for further action.

Updater Transaction

Central to the design of the Updater server is the idea of an Updater transaction. You can add several segment changes to a single Updater Transaction and submit all the changes at once. Each such segment change, addition, or deletion is called an Updater Action.

An Updater transaction is similar to a database transaction. That is, it is atomic. If any segment in the transaction violates a segment entry rule of strict (of type ST) enforcement type the entire transaction is failed. Multiple segment changes are typically submitted in a transaction to ensure that either all or none of the changes are saved. If you do not want all your segment changes if even a single rule violation occurs then you should submit them in different Updater transactions.

Updater Validation Engine

The Updater validation engine is at the heart of all segment validations. It exists both in the Updater client as well as in the Updater server. It performs validations on two types of rules: intrinsic rules and user-defined rules.

Updater Validation: Intrinsic Rules

The Updater client and the Updater server enforce several rules that are not defined by the user. These rules exist to protect data integrity and they cannot be bypassed. That is, all intrinsic rules are of the strict enforcement type. The Updater client enforces some intrinsic rules whereas the Updater server enforces others. If a rule requires additional data to be retrieved from the database it is enforced by the Updater Server, otherwise the Updater Client enforces it.

The following table describes the intrinsic rules enforced by both the Updater server and the Updater client.

Table 11: Intrinsic Rules of the Updater API

Rule #	Category	Description
1	Rule name	**A transaction cannot contain multiple edits of same segment**
	Enforced on	Updater client
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_MULTIPLE_EDITS
	Description	The transaction contains multiple actions for the same segment. This is not allowed since Updater would not know which one to keep and which ones to throw away.
2	Rule name	**The start moment of a segment cannot be greater than its stop moment**
	Enforced on	Updater client
	Applies to	Detail segments only
	Error code	UPDA_RUL_START_GT_STOP

Rule #	Category	Description
	Description	The start moment for the segment cannot be greater than its stop moment—a segment cannot start after its stop time.
3	Rule name	**Segment duration cannot exceed 24 hours**
	Enforced on	Updater client
	Applies to	Detail segments only
	Error code	UPDA_RUL_MAX_SEG_DUR_EXCD
	Description	The segment has a duration greater than 1439 minutes (23 hours, 59 minutes). Segments must be less than 24 hours long.
4	Rule name	**A segment's start or stop moment cannot be outside of the allowed range**
	Enforced on	Updater client
	Applies to	Detail segments only
	Error code	UPDA_RUL_SEG_OUTSIDE_RNG
	Description	The segment's start or stop moment are outside the allowed range for its nominal date. The allowed time range for a segment's start and stop time for a given nominal date is plus or minus 48 hours from midnight (GMT) for the nominal date.
5	Rule name	**The start moment must be valid**
	Enforced on	Updater client
	Applies to	Detail segments only
	Error code	UPDA_RUL_START_MOMENT
	Description	The start moment of the segment is invalid. The start moment of the segment must be in the range from -500000000 to +500000000 (+/- $5x10^8$).
6	Rule name	**The stop moment must be valid**
	Enforced on	Updater client
	Applies to	Detail segments only
	Error code	UPDA_RUL_STOP_MOMENT
	Description	The stop moment of the segment is invalid. The stop moment of the segment must be in the range from -500000000 to +500000000 (+/- $5x10^8$).
7	Rule name	**The nominal date must be valid**
	Enforced on	Updater client

Rule #	Category	Description
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_NOMINAL_DATE
	Description	The nominal date of the segment is invalid. The nominal date of the segment must be in the range from -500000 to +500000 (+/- $5x10^5$).
8	Rule name	**The transaction snapshot must match the database snapshot**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_SNPSHT_VIOL
	Description	The Updater transaction snapshot does not match the database snapshot. This occurs when any of the segments in the transaction has changed after the transaction was submitted and before the transaction was committed to the database.
9	Rule name	**Nominal dates of detail segments cannot be out of order**
	Enforced on	Updater server
	Applies to	Detail segments only
	Error code	UPDA_RUL_NOM_DATE_OUT_OF_ORDER
	Description	The nominal dates of two detail segments are out of order. Nominal dates are considered to be out of order if the stop moment for a segment is greater than the start moment of another segment with a later nominal date. Either both the segments causing the violation are in the transaction, or one of the segment's is an existing segment from the database.
10	Rule name	**Detail segments cannot overlap**
	Enforced on	Updater server
	Applies to	Detail segments only
	Error code	UPDA_RUL_DET_SEG_OVERLAP
	Description	Two detail segments for the same employee, nominal date, and segment code overlap.

Rule #	Category	Description
11	Rule name	**General segments cannot overlap**
	Enforced on	Updater server
	Applies to	General segments only
	Error code	UPDA_RUL_GEN_SEG_OVERLAP
	Description	Two general segments for the same employee, nominal date, and segment code overlap. Either both the segments causing the violation are in the transaction, or one of the segment's is an existing segment from the database.
12	Rule name	**A general segment cannot overlap a detail segment**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_GEN_DET_CODE_OVERLAP
	Description	A general segment is incompatible with a detail segment having the same segment code for the same employee on the same nominal date. Either both the segments causing the violation are in the transaction, or one of the segment's is an existing segment from the database.
13	Rule name	**An employee cannot be inactive**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_EMP_INACTIVE
	Description	The employee for whom the segment is being added or changed has been marked as inactive. Segments cannot be added or changed for inactive employees.
14	Rule name	**An employee must not be terminated**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_EMP_TERMINATED
	Description	The employee for whom the segment is being added or changed has been marked as terminated. The nominal date for which the segment is being added, changed, or deleted is greater than the employee's termination date.

Rule #	Category	Description
15	Rule name	**The nominal date must not be closed out**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_NOM_DATE_CLOSED_OUT
	Description	The nominal date for which the segment is being added has been closed out. No changes can be made to any segments for nominal dates that have been closed out.
16	Rule name	**A memo is required**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_MEMO_REQ
	Description	The segment code assigned for the segment requires that a memo be included with the segment. The segment being added or changed does not have a memo.
17	Rule name	**An employee must have the required skill for a work state**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_SKILL_CHECK
	Description	If the segment being added contributes to a work state then the employee must have a skill for that work state.
18	Rule name	**The user must have permission to add a segment**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_SEG_LOCK_ADD
	Description	The user submitting the transaction must have the permissions to add segments of this segment code.
19	Rule name	**The user must have permission to change the segment code**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_SEG_LOCK_CHANGE_TO
	Description	The user submitting the transaction must have the permissions to change the segment code to the new segment code.

Rule #	Category	Description
20	Rule name	**The user must have permission to modify the segment**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_SEG_LOCK_MODIFY
	Description	The user submitting the transaction must have the permissions to modify segments with this segment code.
21	Rule name	**The user must have permission to delete the segment**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_SEG_LOCK_DELETE
	Description	The user submitting the transaction must have permissions to delete segments with this segment code.
22	Rule name	**The user must have permission to change the segment code**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RULE_SEG_LOCK_CHANGE_FROM
	Description	The user submitting the transaction must have permissions to change the segment code from the existing segment code.
23	Rule name	**Enforce concurrent employee limit**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RULE_EXCEED_EMP_LIMIT
	Description	No segments may be added when the concurrent employee license limit has been exceeded.

Updater Validation: User-Defined Rules

The Updater server also enforces all user-defined rules. The following table describes the two types of rules that users can define.

Table 12: User-Defined Rules of the Updater API

Rule #	Category	Description
1	Rule name	**Segments of a specified category must be contained within segments of another specified category**

Rule #	Category	Description
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_IN
	Description	Segments of category A must be wholly contained within segments of category B.
2	Rule name	**Segments of a specified category must not overlap segments of another specified category**
	Enforced on	Updater server
	Applies to	General segments and detail segments
	Error code	UPDA_RUL_NOV
	Description	Segments of category A must not overlap segments of category B.

About the Updater API

This section describes the Updater API that applications must use when submitting changes to the Updater server. The Updater API consists of several COM interfaces that are used to create transactions, add actions to the transaction, and then submit the changes. The interfaces can also be queried to determine what, if any, violations caused the transaction to fail. Since these interfaces all form part of the Updater client, they are also referred to as the Updater client interfaces.

Basic Programming Model

The Updater API allows you to perform the following sequence of actions:

1. Create an Updater transaction.
2. Add segment changes, additions, or deletions to the transaction (these are called Updater actions).
3. View and make changes to your actions before submitting them.
4. Submit the transaction.
5. Check the status of the transaction.
6. If the transaction did not pass, check the errors and violations that caused the transaction to fail.

You will typically employ all of the above steps when submitting changes to Updater. You could also take advantage of additional functionality provided by the Updater API to exert more control over the way in which your transaction is evaluated.

Updater Programming Model in Detail

The following are the key elements of the Updater API:

- Updater transaction
- Updater actions, consisting of:
- General segment additions
- General segment changes
- General segment deletions
- Detail segment additions
- Detail segment changes
- Detail segment deletions
- Re-summarizing segments
- Updater violations, both warning and strict type violations
- Updater errors

Each element is described in the following sections.

Updater Transaction

The Updater transaction is the means with which applications connect to Updater. You must specify the Aspect Workforce username and password that Updater will use to verify that you are a user of Aspect Workforce. Additionally, changes can also be submitted on behalf of another user to allow you to write server based applications that batch requests from several Aspect Workforce users.

The Updater transaction also allows you to specify the maximum numbers of errors it can encounter before skipping the remainder of validation and failing the transaction. This is allowed so that for a large transaction with a several errors time can be saved since the transaction will fail anyway. You can also specify the priority of the transaction. Higher priority transactions are evaluated before lower priority transactions. Transactions with the same priority are evaluated in the order they are received. You must also specify the name of the computer from where changes are being made as well a name for the application making the changes.

At any time, you can check the status of the transaction. The status of the transaction can be used to determine whether the transaction failed or succeeded. If the transaction failed, a list of all the violations can be obtained for you to take further action. A list of any other errors encountered can also be obtained from the transaction.

Updater Actions

Updater actions are the individual changes that you make to segments. Updater actions can be used to add, delete, or change general and detail segments. Updater actions can also be used to re-summarize the daily segment summary for any employee.

Deletions and changes to segments require that you also supply the existing information for the segment. This is required to ensure that Updater can perform its snapshot validation. If the information you supply as the existing segment properties does not match what is in the database when Updater tries to save the segments, the changes will not be saved and the transaction will be failed. This is to protect the database from unwanted edits by concurrent users.

Updater Violations

Updater violations allow you to examine all the warning-type and strict-type violations encountered by Updater. Updater violations provide detailed information about all the segments involved in a violation, whether the segments are all in the same transaction or existing segments from the database.

Updater Errors

If errors other than segment violations occurred, you get a list of the errors to allow you to rectify the problem.

About the Visual Basic Wrapper

The wrapper files written in Visual Basic version 6 have been removed from this SDK. We strongly encourage you to use the functions included in Alvaria Web Services to submit official schedule changes.

Updater Error Codes

The following tables list the error categories and error codes related to the Updater API.

Table 13: Updater Error Categories

Symbolic Name	Error Code Definition	Number (Hex)	Description
UPDA_INFO	NA	0x1000	Informational message category
UPDA_ERR	NA	0x2000	Updater error category
UPDA_DBERR	NA	0x4000	Updater database error category
UPDA_INVALID_SUBMISSION	NA	0x8000	Invalid submission error category
UPDA_RUL	NA	0x10000	Updater rule violation category
UPDA_UNKNOWN	NA	0x80000000	Unexpected error category
UPDA_SEG_CAT_RUL	UPDA_RUL+0	0x10000	Segment category rule violation category
UPDA_INTRINSIC_RULE	UPDA_RUL+5	0x10005	Intrinsic rule violation category
UPDA_PLUGGABLE_RULE	NA	0x20000	Updater plug-in rule violation category

Symbolic Name	Error Code Definition	Number (Hex)	Description
UPDA_RULE_SEG_LOCK_CHANGE_FROM	UPDA_INTRINSIC_RULE+22	0x1001B	The user submitting the transaction must have permissions to change the segment code from the existing segment code
UPDA_RULE_EXCEED_EMP_LIMIT	UPDA_INTRINSIC_RULE+23	0x1001C	No segments may be added when the concurrent employee license limit has been exceeded

Table 14: Updater Error Codes

Symbolic Name	Error Code Definition	Number (Hex)	Description
UPDA_TRAN_FAILED	UPDA_ERR+1	0x1001	The Updater transaction failed. See the Errors and Violations property of the transaction for details
UPDA_ERR_NO_PROCESS_DB_ALIAS	UPDA_ERR+2	0x1002	The Updater server is not configured to connect to the BDE alias specified in the transaction
UPDA_ERR_CLIENT_NOT_CONNECTED	UPDA_ERR+3	0x1003	The Updater client could not connect to the Updater server. Either the Updater server service is not running or the Updater client is not configured to connect to connect to the machine where the Updater server is running
UPDA_ERR_INVALID_LOGIN_SK	UPDA_ERR+4	0x1004	The transaction does not contain a valid TCS Login SK
UPDA_ERR_UNABLE_TO_LOG_IN	UPDA_ERR+5	0x1005	The Updater server cannot login to the database. Ask your Aspect Workforce administrator to check the Updater server configuration
UPDA_ERR_EXCEED_TIMEOUT	UPDA_ERR+6	0x1006	The Updater server timed out when trying to connect to the database

Symbolic Name	Error Code Definition	Number (Hex)	Description
UPDA_ERR_NULL_MODUL_USER	UPDA_INVALID_SUBMISSION+1	0x8001	The transaction contains a blank user name
UPDA_ERR_NULL_PROG	UPDA_INVALID_SUBMISSION+2	0x8002	The transaction contains a blank program name
UPDA_ERR_NULL_MODUL	UPDA_INVALID_SUBMISSION+3	0x8003	The transaction contains a blank module name
UPDA_ERR_NULL_COMPID	UPDA_INVALID_SUBMISSION+4	0x8004	The transaction contains a blank computer name
UPDA_ERR_NULL_DBALIAS	UPDA_INVALID_SUBMISSION+5	0x8005	The transaction contains a blank database alias
UPDA_ERR_SEG_CODE_BADKEY	UPDA_INVALID_SUBMISSION+6	0x8006	The action contains an invalid segment code SK
UPDA_ERR_EMP_BADKEY	UPDA_INVALID_SUBMISSION+7	0x8007	The action contains an invalid employee SK
UPDA_DBERR_SEG_BADKEY	UPDA_DBERR+1	0x4001	The action contains an invalid segment SK
UPDA_RUL_IN	UPDA_SEG_CAT_RUL+1	0x10001	The action violates a must be contained in rule
UPDA_RUL_NOV	UPDA_SEG_CAT_RUL+2	0x10002	The action violates a must not overlap rule
UPDA_RUL_MULTIPLE_EDITS	UPDA_INTRINSIC_RULE+1	0x10006	There are multiple edits for this segment in the transaction
UPDA_RUL_START_GT_STOP	UPDA_INTRINSIC_RULE+2	0x10007	The segment start is greater than its stop
UPDA_RUL_MAX_SEG_DUR_EXCD	UPDA_INTRINSIC_RULE+3	0x10008	The segment duration is longer than the maximum allowed duration
UPDA_RUL_SEG_OUTSIDE_RNG	UPDA_INTRINSIC_RULE+4	0x10009	The segment is outside of the valid range for the nominal date
UPDA_RUL_START_MOMENT	UPDA_INTRINSIC_RULE+5	0x1000A	The action contains an invalid start moment

Symbolic Name	Error Code Definition	Number (Hex)	Description
UPDA_RUL_STOP_MOMENT	UPDA_INTRINSIC_RULE+6	0x1000B	The action contains an invalid stop moment
UPDA_RUL_NOMINAL_DATE	UPDA_INTRINSIC_RULE+7	0x1000C	The action contains an invalid nominal date
UPDA_RUL_SNPSHT_VIOL	UPDA_INTRINSIC_RULE+8	0x1000D	The transaction snapshot does not match the database snapshot
UPDA_RUL_NOM_DATE_OUT_OF_ORDER	UPDA_INTRINSIC_RULE+9	0x1000E	The nominal dates of the segments are out of order
UPDA_RUL_DET_SEG_OVERLAP	UPDA_INTRINSIC_RULE+10	0x1000F	Two details segments overlap
UPDA_RUL_GEN_SEG_OVERLAP	UPDA_INTRINSIC_RULE+11	0x10010	Two general segments overlap
UPDA_RUL_GEN_DET_CODE_OVERLAP	UPDA_INTRINSIC_RULE+12	0x10011	A general segment overlaps a detail segment
UPDA_RUL_EMP_INACTIVE	UPDA_INTRINSIC_RULE+13	0x10012	The employee in this segment has been marked inactive
UPDA_RUL_EMP_TERMINATED	UPDA_INTRINSIC_RULE+14	0x10013	The segment's nominal date is past the employee's termination date
UPDA_RUL_NOM_DATE_CLOSED_OUT	UPDA_INTRINSIC_RULE+15	0x10014	The segment is for a nominal date that has already been closed out
UPDA_RUL_MEMO_REQ	UPDA_INTRINSIC_RULE+16	0x10015	This segment code requires a memo with the segment
UPDA_RUL_SKILL_CHECK	UPDA_INTRINSIC_RULE+17	0x10016	The employee does not have the skill for the work state associated with the segment
UPDA_RUL_SEG_LOCK_ADD	UPDA_INTRINSIC_RULE+18	0x10017	The user submitting the transaction does not have the permissions to add a segment of this type
UPDA_RUL_SEG_LOCK_CHANGE_TO	UPDA_INTRINSIC_RULE+19	0x10018	The user submitting the transaction does not have the permissions to change the segment code to this segment code

Symbolic Name	Error Code Definition	Number (Hex)	Description
UPDA_RUL_SEG_LOCK_MODIFY	UPDA_INTRINSIC_RULE +20	0x10019	The user submitting the transaction does not have the permissions to modify this segment
UPDA_RUL_SEG_LOCK_DELETE	UPDA_INTRINSIC_RULE +21	0x1001A	The user submitting the transaction does not have the permissions to delete the segment
UPDA_RULE_SEG_LOCK_CHANGE_FROM	UPDA_INTRINSIC_RULE +22	0x1001B	The user submitting the transaction must have permissions to change the segment code from the existing segment code
UPDA_RULE_EXCEED_EMP_LIMIT	UPDA_INTRINSIC_RULE +23	0x1001C	No segments may be added when the concurrent employee license limit has been exceeded
UPDA_ERR_RULE_TIMEOUT	UPDA_ERR + 7	0x1007	Updater timed out calling a plug-in validation rule

COM Interfaces Reference

Before you can use the Updater API you will need to add the Updater Client and the UpdaterCommon type libraries to your project. If your are using Visual Basic to access the Updater API, you can do this by selecting Project from the main menu and then checking the box next to Updater Client Type Library and Updater types common to Updater client and Updater server. If the Updater type libraries are not visible in References dialog, you might not have Aspect Workforce installed on your computer. If you are using another development environment, please consult the documentation accompanying that product regarding how to include references to type libraries in your project.

To browse the objects in the Updater type libraries using Visual Basic, choose View from the menu and then select Object Browser. Then, from the drop-down list, select UpdaterClient. This will show you all the objects in the UpdaterClient library.

All the interfaces in the Updater API are dual interfaces that are derived from IDispatch. Therefore, in addition to the methods listed here, all the methods of IDispatch are also available.

IUpdaterTransaction

The IUpdaterTransaction interface enables you to submit changes to Updater. This section lists the methods and properties of the IUpdaterTransaction interface along with brief descriptions of each. It also discusses how to implement the interface.

Implementation

The IUpdaterTransaction interface is implemented by the UpdaterTransactionCoClass defined in the UpdaterClient type library. To create an UpdaterTransactionCoClass and get a reference to an IUpdaterTransaction interface, you can do the following:

```
Dim transaction As UpdaterClient.IUpdaterTransaction
Set transaction = New UpdaterClient.UpdaterTransactionCoClass
```

You can then use the transaction variable to add actions to it and submit them to Updater for processing.

Methods (Listed)

The following table lists and briefly describes the methods of the IUpdaterTransaction interface.

Table 15: IUpdaterTransaction Methods

Method	Description
AddUpdaterAction	Adds an Updater action to the transaction and returns a reference to it.
AddStrictViolation	Adds a strict IUpdaterViolation to the transaction and returns a reference to it.
AddWarningViolation	Adds a warning IUpdaterViolation to the transaction and returns a reference to it.
AddError	Adds an IUpdaterError to the transaction and returns a reference to it.
Submit	Submits the transaction for processing

Properties (Listed)

The following table lists and briefly describes the properties of the IUpdaterTransaction interface.

Table 16: IUpdaterTransaction Properties

Property	Description
Status	Returns the status of the transaction.
SequenceNum	Returns the sequence number of the transaction. The sequence number can be used by clients to if there have been other transactions since this one.
MaxErrorCount	Allows you to set the maximum number of violations that can be tolerated before the transaction is aborted.
UpdaterActionCount	Number of Updater actions in transaction.
UpdaterActions	Array property of Updater actions. Returns the action specified by the index.
Priority	Priority of transaction.

Property	Description
UserName	Aspect Workforce username of user submitting transaction.
Password	Aspect Workforce password of user submitting transaction.
BDE_Alias	Database alias.
SourceApplication	Name of application or module submitting the transaction.
ComputerID	Name of computer from where the transaction is being submitted.
StrictViolationsCount	Number of violations of strict-enforcement type. Available after transaction has been evaluated.
StrictViolations	Array property of violations of strict-enforcement type.
WarningViolationsCount	Number of violations of warning-enforcement type. Available after transaction has been evaluated.
WarningViolations	Array property of violations of warning-enforcement type.
ErrorCount	Number of errors encountered in the transaction.
Errors	Array property of violations of errors.
UserKey	Submitting Aspect Workforce user's SK. This is needed to allow submissions on behalf of another user.
ClientLocale	Locale information that should be used by Updater plug-in rules. For more information, see Chapter 3, Updater Plug-In Rule Developer Guide.

Methods (Documented)

This section documents the methods of the IUpdaterTransaction interface.

IUpdaterTransaction::AddUpdaterAction(ActionType As TUActionType)

This method adds an action of the specified type to the transaction and returns a reference to it.

Syntax	`Dim action as UpdaterClient.IUpdaterAction` `Set action = transaction.AddUpdaterAction(`*`ActionType`*`)`
Parameters	*ActionType:* A TUActionType enumeration value that specifies the type of action to add to the transaction. See Table 2-17 for valid action types.

Return values	Returns an IUpdaterAction interface of the type specified in ActionType.
Remarks	The AddUpdaterAction method returns an IUpdaterAction interface of the type specified in ActionType. The returned interface must then be used to fill in the action properties, which further determine which segment is being changed and to what, or which segment is being deleted, or what the properties of the segment being added are. Note that the method returns a generic IUpdaterAction reference and not of the specific type created. You must cast this reference to the appropriate type before using it.

Table 17: TUActionActionType Valid Values

Value	Description
UActionType_None	No action. Represents an unmodified segment. This action type is used by Updater internally. Users typically do not submit unmodified segments to Updater.
UActionType_GsegInsert	Insert a general segment.
UActionType_DsegInsert	Insert a detail segment.
UActionType_GsegUpdate	Update a general segment.
UActionType_DsegUpdate	Update a detail segment.
UActionType_GsegDelete	Delete a general segment.
UActionType_DsegDelete	Delete a detail segment.
UActionType_Summarize	Resummarize superstate hours for an employee and nominal date.

IUpdaterTransaction:: AddStrictViolation (Seg1Key As Double,

Seg1Detail As TBool,

Seg1Employee As Double,

Seg1NomDate As Long,

Seg2Key As Double,

Seg2Detail As TBool,

Seg2Employee As Double,

Seg2NomDate As Long,

ErrorCode As Long,

ErrorMessage As BSTR)

This method adds a strict IUpdaterViolation to the transaction and returns a reference to it.

Syntax	Dim action as UpdaterClient.IUpdaterViolation
	`Set action = transaction.`
	`AddStrictViolation`
	`(Seg1Key,`
	`Seg1Detail,`
	`Seg1EmployeeKey,`
	`Seg1NominalDate,`
	`Seg2Key,`
	`Seg2Detail,`
	`Seg2EmployeeKey,`
	`Seg2NominalDate,`
	`ErrorCode,`
	`ErrorMessage)`

Parameters	(See IUpdaterViolation for a description of the parameters.)
Return values	Returns an IUpdaterViolation interface.
Remarks	The AddUpdaterAction method returns an IUpdaterAction interface of the type specified in ActionType. The returned interface must then be used to fill in the action properties, which further determine which segment is being changed and to what, or which segment is being deleted, or what the properties of the segment being added are. Note that the method returns a generic IUpdaterAction reference and not of the specific type created. You must cast this reference to the appropriate type before using it.

IUpdaterTransaction:: AddWarningViolation (Seg1Key As Double,

Seg1Detail As TBool,

Seg1Employee As Double,

Seg1NomDate As Long,

Seg2Key As Double,

Seg2Detail As TBool,

Seg2Employee As Double,

Seg2NomDate As Long,

ErrorCode As Long,

ErrorMessage As BSTR)

This method adds a warning IUpdaterViolation to the transaction and returns a reference to it.

Syntax	Dim action as UpdaterClient.IUpdaterViolation Set action = transaction. AddWarningViolation (*Seg1Key*, *Seg1Detail*, *Seg1EmployeeKey*, *Seg1NominalDate*, *Seg2Key*, *Seg2Detail*, *Seg2EmployeeKey*, *Seg2NominalDate*, *ErrorCode*, *ErrorMessage)*
Parameters	(See IUpdaterViolation for a description of the parameters.
Return values	Returns an IUpdaterViolation interface.
Remarks	(See IUpdaterViolation for remarks about the parameters.)

IUpdaterTransaction::AddError (ErrorCode As Long, ErrorMessage As BSTR)

This method adds an IUpdaterError to the transaction and returns a reference to it.

Syntax	Dim action as UpdaterClient.IUpdaterViolation `Set action = transaction. AddError (ErrorCode,ErrorMessage)`
Parameters	*ErrorCode:* A Long value that indicates the error number. *ErrorMessage:* A message String that pertains to the error.
Return values	Returns an IUpdaterError interface.
Remarks	Typically, this function is used by Updater plug-in rules. For that use, the value of ErrorCode would be assigned by the plug-in rule developer and should be in the range of 0 to 4096. The contents of ErrorMessage may or may not be localized at the discretion of the plug-in rule developer.

IUpdaterTransaction::Submit

This method submits the transaction to Updater.

Syntax	transaction.Submit
Parameters	None.
Return values	None.
Remarks	The Submit method submits the transaction for processing. Once the transaction has been submitted, it cannot be cancelled. To abort a transaction without submitting it simply set the transaction variable to Nothing to remove all references to it.

Properties (Documented)

IUpdaterTransaction::Status

This property indicates the status of a transaction.

Settings/ Return values	This is a read-only property that returns a TUStatus enumeration value that indicates the status of the transaction.
Remarks	Use the Status property of IUpdaterTransaction to determine its status at any time. See TUStatus for possible values of Status and their meaning.

IUpdaterTransaction::SequenceNum

This property indicates the sequence number of the transaction.

Settings/ Return values	This is a read-only property that returns the sequence number of the transaction as a Double.
Remarks	The SequenceNum property is used by other Aspect Workforce modules that use Updater for segment notification but also get segment data from the database. It is used to calculate whether the notification they received is for the most current Updater transaction that has been saved in to the database. Most users of the Updater API will not need to use this property.

IUpdaterTransaction::UpdaterActionCount

This property indicates the number of actions in the transaction.

Settings/ Return values	This is a read-only Long property that returns the number of Updater actions in the transaction.
Remarks	Use the UpdaterActionCount property to determine the number of actions in the transaction. You can use this value to enumerate the actions and make further changes to them before submitting the transaction.

IUpdaterTransaction::UpdaterActions(Index As Long)

This property returns the Updater action referenced by *Index.*

Syntax	`Dim action As UpdaterClient.IUpdaterAction`
	
	`Set action = transaction.UpdaterActions(0)`

Parameters	*Index:* A Long value that indicates the index of the action to return.
Settings/ Return values	This is a read-only property that returns an IUpdaterAction interface stored at the index indicated by Index.
Remarks	The number of actions in the transaction can be determined by the UpdaterActionCount property. The index is zero based. Therefore, the first action is at index 0 and the last at UpdaterActionCount-1.

IUpdaterTransaction::Priority

This property indicates the priority of the transaction.

Settings/ Return values	Sets or returns a TUPriority enumeration value that indicates the priority of the transaction. The default value is UPriority_Normal.
Remarks	Use the Priority property to set the priority of a transaction. Updater queues all transactions first by their Priority and then in the order in which they are received. Therefore, a higher Priority transaction will get queued above one with a lower Priority. Note, however, that a transaction in the process of being evaluated cannot be pre-empted by transactions of higher priority. A higher Priority simply queues the transaction above other lower Priority transactions that are in queue.

IUpdaterTransaction::UserName

This property indicates the Aspect Workforce username of the user submitting the transaction.

Settings/ Return values	Sets or returns a String value that indicates the Aspect Workforce username of the user submitting the transaction. The default value is a blank string.
Remarks	Use the UserName property to tell Updater the username of the submitting user. You must set this to a valid Aspect Workforce username otherwise the transaction will fail.

IUpdaterTransaction::Password

This property indicates the password for the Aspect Workforce username of the user submitting the transaction.

Settings/ Return values	Sets or returns a String value that indicates the password for the Aspect Workforce username of the user submitting the transaction. The default value is a blank string.
Remarks	Use the Password property to tell Updater the password of the submitting user. You must set this to the correct password for the user otherwise the transaction will fail.

IUpdaterTransaction::BDE_Alias

This property indicates the alias of the database that the transaction should submit changes to.

Settings/ Return values	Sets or returns a String value that indicates the name of the database you are submitting the segment changes for. The default value is blank string.
Remarks	You must specify the database alias where Updater should write its changes. Note that this is specific to the way in which Aspect Workforce is set up and not the way in which any other data sources may be configured. You can obtain the value you should set *BDE_Alias* to by running the Aspect Workforce client. The database name shown in the login dialog is the value that you need to use for *BDE_Alias*.

IUpdaterTransaction::SourceApplication

This property indicates the name of the application or module submitting the transaction.

Settings/ Return values	Sets or returns a String value that indicates the name of the application or module submitting the transaction. The default value is a blank string.
Remarks	The SourceApplication property is used by Updater to log the name of the module that sent the transaction. This is useful for tracking where the changes came from. You must set this to a value that is not a blank string otherwise the transaction will fail.

IUpdaterTransaction::ComputerID

This property indicates the name of the computer that submitted the transaction.

Settings/ Return values	Sets or returns a String value that indicates the name of the computer from where the transaction was submitted.
Remarks	The ComputerID property is used by Updater to log the name of the computer from where the transaction was submitted. This is useful for tracking where the changes came from. You must set this to a value that is not a blank string otherwise the transaction will fail.

IUpdaterTransaction::StrictVolationsCount

This property indicates the number of strict-enforcement type violations encountered in the transaction.

Settings/ Return values	Read-only Long property that returns the number of strict-enforcement type violations in the transaction.
Remarks	Use the StrictViolationsCount property to determine the number of strict-enforcement type violations in the transaction. You can use this value to enumerate the violations to determine why the transaction failed.

IUpdaterTransaction::StrictViolations(Index As Long)

This property returns the strict-enforcement type violation referenced by *Index*.

Syntax	`Dim violation As UpdaterClient.IViolation` `Set violation = transaction.StrictViolations(0)`
Parameters	*Index*: A Long value that indicates the index of the violation to return.
Settings/ Return values	This is a read-only property that returns an IUpdaterViolation interface stored at the index indicated by Index.
Remarks	The number of strict-enforcement type violations in the transaction can be determined by the StrictViolationsCount property. The index is zero based. Therefore, the first violation is at index 0 and the last at StrictViolationsCount-1.

IUpdaterTransaction::WarningVolationsCount

This property indicates the number of warning-enforcement type violations encountered in the transaction.

Settings/ Return values	Read-only Long property that returns the number of warning-enforcement type violations in the transaction.
Remarks	Use the WarningViolationsCount property to determine the number of warning-enforcement type violations in the transaction. You can use this value to enumerate the warnings violations generated by the transaction.

IUpdaterTransaction::WarningViolations(Index As Long)

This property returns the warning-enforcement type violation referenced by *Index*.

Syntax	`Dim violation As UpdaterClient.IViolation` `Set violation = transaction.WarningViolations(0)`
Parameters	*Index*: A Long value that indicates the index of the violation to return.

Settings/ Return values	This is a read-only property that returns an IUpdaterViolation interface stored at the index indicated by *Index*.
Remarks	The number of warning-enforcement type violations in the transaction can be determined by the WarningViolationsCount property. The index is zero based. Therefore, the first violation is at index 0 and the last at WarningViolationsCount-1.

IUpdaterTransaction::ErrorCount

This property indicates the number of errors encountered in the transaction.

Settings/ Return values	Read-only Long property that returns the number of errors encountered in the transaction.
Remarks	Use the ErrorCount property to determine the number of errors in the transaction. You can use this value to enumerate the errors generated by the transaction

IUpdaterTransaction::Errors(Index As Long)

This property returns the error referenced by Index.

Syntax	`Dim error As UpdaterClient.IUpdaterError`
	
	`Set error = transaction.Errors(0)`

Parameters	*Index*: A Long value that indicates the index of the error to return.
Settings/ Return values	This is a read-only property that returns an IUpdaterError interface stored at the index indicated by *Index*.
Remarks	The number of errors in the transaction can be determined by the ErrorCount property. The index is zero based. Therefore, the first violation is at index 0 and the last at ErrorCount-1.

IUpdaterTransaction::UserKey

This property indicates TCS_LOGIN_SK of the user submitting the transaction.

Settings/ Return values	Sets or returns a Double value that indicates the TCS_LOGIN_SK (from the TCS_LOGIN table) of the user submitting the transaction.

Remarks	Use the UserKey property to specify the surrogate key of the Aspect Workforce user submitting the transaction. Typically, this surrogate key corresponds to the same user referenced by the UserName property. However, if the transaction is being submitted by one Aspect Workforce user on behalf of another, such as in an application that processes agent schedule change requests, then the UserName and Password properties should correspond to the Aspect Workforce user running the application and the UserKey property should be that of the user on whose behalf the transaction is being submitted. You must specify a valid surrogate key for a user in the Aspect Workforce database. Otherwise the transaction fails.

IUpdaterTransaction::ClientLocale

This property indicates locale information that should be used by Updater plug-in rules.

Settings/ Return values	Sets or returns a BSTR value that contains locale information.
Remarks	The intended purpose of this property is to pass locale information to Updater plug-in rules. If Updater plug-in rules are not being used, then there is no need to set this parameter. If Updater plug-in rules are being used, then setting this property is still optional. If it is not set (i.e., ClientLocale is an empty string), then Updater Client (i.e., UpdaterClient.dll) would populate ClientLocale with information based on the locale of the machine running Updater Client. Therefore, the Client Application would only set this field if the information that would be set by Updater Client was not acceptable. See Locale Data Format for a description of the format of the locale data.

IUAGsegInsert

The IUAGsegInsert interface enables you to add a general segment.

Implementation

The IUAGsegInsert interface is implemented by the UAGsegInsertCoClass defined in the UpdaterClient type library. However, you cannot create an instance of a UAGsegInsertCoClass directly. You must choose to add an action to a transaction and choose the appropriate TUActionType.For example, you can do the following to add a general segment:

```
Dim action As UpdaterClient.IUAGsegInsert

Set action = transaction.AddUpdaterAction(UActionType_GSegInsert)
```

You can then modify the properties of action variable to the properties of the general segment you want to add.

Properties (Listed)

The following table lists and briefly describes the properties of the IUAGsegInsert interface.

Table 18: IUAGsegInsert Properties

Property	Description
SaveMemo	Whether or not the memo should be saved.
ActionType	Type of action.
Action_IIDStr	String representation of GUID that represents the action.
ProgID	Name of module that submitted the action.
New_Seg_SK	SK for segment to be added.
New_Emp_SK	Employee SK assigned to segment.
New_Nom_Date	Nominal date of segment being added.
New_Seg_Code_SK	Segment code SK assigned to segment being added.
New_Memo	Memo for segment.

Properties (Documented)

IUAGsegInsert::SaveMemo

This property indicates whether the memo specified in Memo property should be saved.

Settings/ Return values	Sets and returns a TBool value that specifies whether the memo specified in the Memo property should be saved to the database.
Remarks	The SaveMemo property is used to indicate to Updater whether it save the memo specified in the Memo property. This property applies only to actions that *modify* general and detail segments. This property allows you to not have to retrieve the memo for those actions if you are not going to change it. For example, when modifying either a general or a detail segment if you are not going to change the existing memo for the segment, you can set SaveMemo to false. That will save all the other changes you make for the segment but preserve the existing memo, if any. *For all other actions, this property is ignored.*

IUAGsegInsert::ActionType

This property indicates what kind of action this is.

Settings/ Return values	Read-only property of TUActionType that indicates what kind of action is to be performed.

Remarks	The value of ActionType is determined when the action is created by a call to IUpdaterTransaction.AddUpdaterAction.

IUAGsegInsert::Action_IIDStr

This is a read-only property. It returns the interface ID of the interface in registry string format.

IUAGsegInsert::ProgID

This property indicates the program or module that submitted this action.

Settings/ Return values	Returns a String property that indicates the name of the program or module that submitted this action.
Remarks	Read-only property that returns a String indicating the name of the program or module that submitted this action.

IUAGsegInsert::New_Seg_SK

This property indicates the SK for the segment being inserted.

Settings/ Return values	Sets and returns a Double value that indicates the SK for segment being inserted.
Remarks	You must supply a SK for the segment you are inserting. Note that although the type of this property is Double, it is defined in the database as a NUMERIC(15,1) value. Therefore, any number that has any digits after the decimal point is invalid. You should only use SKs generated by a call to either the GET_SK or GET_SKS stored procedures in the database.

IUAGsegInsert::New_Emp_SK

This property indicates the SK of the employee being assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the employee being assigned to this segment.
Remarks	Set this property to the SK of the employee (EMP_SK from the EMP table) to whom you are assigning this segment.

IUAGsegInsert::New_Nom_Date

This property indicates the nominal date for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the nominal date for the segment.
Remarks	Set this property to the nominal date to which you wish to add this segment.

IUAGsegInsert::New_Seg_Code_SK

This property indicates the SK of the segment code assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the segment code being assigned to this segment.
Remarks	Set this property to the SK of segment code (SEG_CODE_SK from the SEG_CODE) that you wish to assign to this segment.

IUAGsegInsert::New_Memo

This property indicates the memo for this segment.

Settings/ Return values	Sets and returns a String value that indicates the memo for this segment.
Remarks	You can add a memo to any segment that you insert. Some segment codes require you to supply a memo. If the segment code specified in New_Seg_Code_SK is such a segment code, you must supply a string that is not blank or the transaction will fail.

IUAGsegUpdate

The IUAGsegUpdate interface enables you to modify a general segment.

Implementation

The IUAGsegUpdate interface is implemented by the UAGsegUpdateCoClass defined in the UpdaterClient type library. However, you cannot create an instance of a UAGsegUpdateCoClass directly. You must choose to add an action to a transaction and choose the appropriate TUActionType.For example, you can do the following to add an action to modify a general segment:

```
Dim action As UpdaterClient.IUAGsegUpdate
Set action = transaction.AddUpdaterAction(UActionType_GSegUpdate)
```

You can then modify the properties of action variable to change the general segment.

Properties (Listed)

The following table describes the properties of the IUAGsegUpdate interface.

Table 19: IUAGsegUpdate Properties

Property	Description
SaveMemo	Whether or not the memo should be saved.
ActionType	Type of action.
Action_IIDStr	String representaion of GUID that represents the action.
ProgID	Name of module that submitted the action.
Old_Seg_SK	SK for segment being modified.
Old _Emp_SK	SK of employee originally assigned to segment.

Old _Nom_Date	Original Nominal date of segment.
Old _Seg_Code_SK	SK for original segment code SK assigned to segment.
Old _Memo	Original memo for segment.
New_Seg_SK	SK for segment being modified.
New_Emp_SK	SK of employee SK being assigned to segment.
New_Nom_Date	New nominal date of segment.
New_Seg_Code_SK	SK of new segment code SK being assigned to segment.
New_Memo	New memo for segment.

Properties (Documented)

IUAGsegUpdate::SaveMemo

This property indicates whether the memo specified in Memo property should be saved.

Settings/ Return values	Sets and returns a TBool value that specifies whether the memo specified in the Memo property should be saved to the database.
Remarks	The SaveMemo property is used to indicate to Updater whether it save the memo specified in the Memo property. This property applies only to actions that *modify* general and detail segments. This property allows you to not have to retrieve the memo for those actions if you are not going to change it. For example, when modifying either a general or a detail segment if you are not going to change the existing memo for the segment, you can set SaveMemo to false. That will save all the other changes you make for the segment but preserve the existing memo, if any. *For all other actions, this property is ignored.*

IUAGsegUpdate::ActionType

This property indicates what kind of action this is.

Settings/ Return values	Read-only property of TUActionType that indicates what kind of action is to be performed.
Remarks	The value of ActionType is determined when the action is created by a call to IUpdaterTransaction.AddUpdaterAction.

IUAGsegUpdate::Action_IIDStr

This is a read-only property. It returns the interface ID of the interface in registry string format.

IUAGsegUpdate::ProgID

This property indicates the program or module that submitted this action.

Settings/ Return values	Returns a String property that indicates the name of the program or module that submitted this action.
Remarks	Read-only property that returns a String indicating the name of the program or module that submitted this action.

IUAGsegUpdate::Old_Seg_SK

This property indicates the SK for the segment being modified.

Settings/ Return values	Sets and returns a Double value that indicates the SK for segment being modified.
Remarks	Set this property to the value of the SK of the segment you wish to modify.

IUAGsegUpdate::Old_Emp_SK

This property indicates the SK for the segment being modified.

Settings/ Return values	Indicates the SK of the employee originally assigned to this segment.
Remarks	Set this property to the SK of the employee (EMP_SK from the EMP table) that was originally assigned to this segment. To change the employee assigned to this segment, change New_Emp_SK to the SK of the new employee.

IUAGsegUpdate::Old_Nom_Date

This property indicates the original nominal date for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the original nominal date of the segment.
Remarks	Set this property to the original nominal date of the segment.

IUAGsegUpdate::Old_Seg_Code_SK

This property indicates the SK of the original segment code assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the segment code originally assigned to the segment.
Remarks	You must set this property to the SK of segment code (SEG_CODE_SK from the SEG_CODE) originally assigned to the segment.

IUAGsegUpdate::Old_Memo

This property indicates the original memo for the segment.

Settings/ Return values	Sets and returns a String value that indicates the original memo for the segment.

Remarks	Set this value to the original memo for the segment that you retrieve from the database. You must do this if you wish to change the memo for the segment, which can be done by setting the New_Memo property to the value you wish to change the memo to. If you wish to preserve the old memo, set SaveMemo to Bool_False and leave this property blank.

IUAGsegUpdate::New_Seg_SK

This property indicates the SK for the segment being modified.

Settings/ Return values	Sets and returns a Double value that indicates the SK for segment being modified.
Remarks	For actions that modify a segment, this property is linked to the Old_Seg_SK property, which is used to select the segment being modified. Do not change this value. Use the Old_Seg_SK property instead.

IUAGsegUpdate::New_Emp_SK

This property Indicates the SK of the employee being assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the employee being assigned to this segment.
Remarks	Set this property to the SK of the employee (EMP_SK from the EMP table) to whom you are assigning this segment. If you are changing the employee assigned to this segment, change it from the value that was retrieved from the database, which must be set in Old_Emp_SK.

IUAGsegUpdate::New_Nom_Date

This property indicates the nominal date for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the nominal date that the segment is being changed to.
Remarks	Set this property to the nominal date to which you wish to change this segment to.

IUAGsegUpdate::New_Seg_Code_SK

This property Indicates the SK of the segment code assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the segment code that the segment is being changed to.
Remarks	Set this property to the SK of the segment code (SEG_CODE_SK from the SEG_CODE) that you wish to change this segment to.

IUAGsegUpdate::New_Memo

This property Indicates the memo for this segment.

Settings/ Return values	Sets and returns a String value that indicates the memo for this segment.
Remarks	Set this property to the new value you wish to have for the memo of the segment. If you wish to preserve the old memo of the segment, you can leave this property blank if you also set SaveMemo to Bool_False.

IUAGsegDelete

The IUAGsegDelete interface enables you to delete a general segment.

Implementation

The IUAGsegDelete interface is implemented by the UAGsegDeleteCoClass defined in the UpdaterClient type library. However, you cannot create an instance of a UAGsegDeleteCoClass directly. You must choose to add an action to a transaction and choose the appropriate TUActionType. For example, you can do the following to add an action to modify a general segment:

```
Dim action As UpdaterClient.IUAGsegDelete
Set action = transaction.AddUpdaterAction(UActionType_GSegDelete)
```

You can then modify the properties of action variable to the delete the general segment.

Properties (Listed)

The following table describes the properties of the IUAGsegDelete interface.

Table 20: IUAGsegDelete Properties

Property	Description
SaveMemo	This property is not used for this action.
ActionType	Type of action.
Action_IIDStr	String representaion of GUID that represents the action.
ProgID	Name of module that submitted the action.
Old_Seg_SK	SK for segment being deleted.
Old _Emp_SK	SK of employee assigned to segment being deleted.
Old _Nom_Date	Nominal date of segment being deleted
Old _Seg_Code_SK	SK for segment code SK assigned to segment being deleted.
Old _Memo	Memo of segment being deleted.

Properties (Documented)

IUAGsegDelete::SaveMemo

This property indicates whether the memo specified in Memo property should be saved.

Settings/ Return values	NA
Remarks	This property is not used for this action.

IUAGsegDelete::ActionType

This property indicates what kind of action this is.

Settings/ Return values	Read-only property of TUActionType that indicates what kind of action is to be performed.
Remarks	The value of ActionType is determined when the action is created by a call to IUpdaterTransaction.AddUpdaterAction.

IUAGsegDelete::Action_IIDStr

This is a read-only property. It returns the interface ID of the interface in registry string format.

IUAGsegDelete::ProgID

This property indicates the program or module that submitted this action.

Settings/ Return values	Returns a String property that indicates the name of the program or module that submitted this action.
Remarks	Read-only property that returns a String indicating the name of the program or module that submitted this action.

IUAGsegDelete::Old_Seg_SK

This property indicates the SK for the segment being deleted.

Settings/ Return values	Sets and returns a Double value that indicates the SK for segment being deleted.
Remarks	Set this property to the value of the SK of the segment you wish to delete.

IUAGsegDelete::Old_Emp_SK

This property indicates the SK of the employee assigned to the segment being deleted.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the employee assigned to the segment being deleted.
Remarks	Set this property to the SK of the employee (EMP_SK from the EMP table) assigned to the segment being deleted.

IUAGsegDelete::Old_Nom_Date

This property indicates the nominal date of the segment being deleted.

Settings/ Return values	Sets and returns a Long value that indicates the nominal date of the segment being deleted.
Remarks	Set this property to the nominal date of the segment being deleted.

IUAGsegDelete::Old_Seg_Code_SK

This property indicates the SK of the segment code assigned to the segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the segment code assigned to the segment.
Remarks	You must set this property to the SK of segment code (SEG_CODE_SK from the SEG_CODE) assigned to the segment being deleted.

IUAGsegDelete::Old_Memo

This property indicates the memo for the segment being deleted.

Settings/ Return values	Sets and returns a String value that indicates the memo for the segment.
Remarks	Set this value to the original memo for the segment that you retrieve from the database. You must do this to delete the segment.

IUADsegInsert

The IUADsegInsert interface enables you to add a detail segment.

Implementation

The IUADsegInsert interface is implemented by the UADsegInsertCoClass defined in the UpdaterClient type library. However, you cannot create an instance of a UADsegInsertCoClass directly. You must choose to add an action to a transaction and choose the appropriate TUActionType.For example, you can do the following to add a detail segment:

```
Dim action As UpdaterClient.IUADsegInsert
Set action = transaction.AddUpdaterAction(UActionType_DSegInsert)
```

You can then modify the properties of the action variable to the properties of the detail segment you want to add.

Properties (Listed)

The following table describes the properties of the IUADsegInsert interface.

Table 21: IUADsegInsert Properties

Property	Description
SaveMemo	Whether or not the memo should be saved.
ActionType	Type of action.

Property	Description
Action_IIDStr	String representaion of GUID that represents the action.
ProgID	Name of module that submitted the action.
New_Seg_SK	SK for segment to to added.
New_Emp_SK	Employee SK assigned to segment.
New_Nom_Date	Nominal date of segment being added.
New_Seg_Code_SK	Segment code SK assigned to segment being added.
New_Memo	Memo for segment.
New_Start_Moment	Start moment for segment being added.
New_Stop_Moment	Stop moment for segment being added.

Properties (Documented)

IUADsegInsert::SaveMemo

This property indicates whether the memo specified in Memo property should be saved.

Settings/ Return values	Sets and returns a TBool value that specifies whether the memo specified in the Memo property should be saved to the database.
Remarks	The SaveMemo property is used to indicate to Updater whether it save the memo specified in the Memo property. This property applies only to actions that *modify* general and detail segments. This property allows you to not have to retrieve the memo for those actions if you are not going to change it. For example, when modifying either a general or a detail segment if you are not going to change the existing memo for the segment, you can set SaveMemo to false. That will save all the other changes you make for the segment but preserve the existing memo, if any. *For all other actions, this property is ignored.*

IUADsegInsert::ActionType

This property indicates what kind of action this is.

Settings/ Return values	Read-only property of TUActionTypethat indicates what kind of action is to be performed.
Remarks	The value of ActionType is determined when the action is created by a call to IUpdaterTransaction.AddUpdaterAction.

IUADsegInsert::Action_IIDStr

This is a read-only property. It returns the interface ID of the interface in registry string format.

IUADsegInsert::ProgID

This property indicates the program or module that submitted this action.

Settings/ Return values	Returns a String property that indicates the name of the program or module that submitted this action.
Remarks	Read-only property that returns a String indicating the name of the program or module that submitted this action.

IUADsegInsert::New_Seg_SK

This property indicates the SK for the segment being inserted.

Settings/ Return values	Sets and returns a Double value that indicates the SK for segment being inserted.
Remarks	You must supply a SK for the segment you are inserting. Note that although the type of this property is Double, it is defined in the database as a NUMERIC(15,1) value. Therefore, any number that has any digits after the decimal point is invalid. You should only use SKs generated by a call to either the GET_SK or GET_SKS stored procedures in the database.

IUADsegInsert::New_Emp_SK

This property indicates the SK of the employee being assigned to the segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the employee being assigned to the segment.
Remarks	Set this property to the SK of the employee (EMP_SK from the EMP table) to whom you are assigning the segment.

IUADsegInsert::New_Nom_Date

This property indicates the nominal date for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the nominal date for the segment.
Remarks	Set this property to the nominal date to which you wish to add this segment.

IUADsegInsert::New_Seg_Code_SK

This property indicates the SK of the segment code assigned to the segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the segment code being assigned to the segment.
Remarks	Set this property to the SK of segment code (SEG_CODE_SK from the SEG_CODE) that you wish to assign to the segment.

IUADsegInsert::New_Memo

This property indicates the memo for this segment.

Settings/ Return values	Sets and returns a String value that indicates the memo for this segment.
Remarks	You can add a memo to any segment that you insert. Some segment codes require you to supply a memo. If the segment code specified in New_Seg_Code_SK is such a segment code, you must supply a string that is not blank or the transaction will fail.

IUADsegInsert::New_Start_Moment

This property indicates the start moment for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the start moment for the segment.
Remarks	Set the New_Start_Moment to the start moment you want to assign to the segment.

IUADsegInsert::New_Stop_Moment

This property indicates the stop moment for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the stop moment for the segment.
Remarks	Set the New_Stop_Moment to the stop moment you want to assign to the segment.

IUADsegUpdate

The IUADsegUpdate interface enables you to modify a detail segment.

Implementation

The IUADsegUpdate interface is implemented by the UADsegUpdateCoClass defined in the UpdaterClient type library. However, you cannot create an instance of a UADsegUpdateCoClass directly. You must choose to add an action to a transaction and choose the appropriate TUActionType. For example, you can do the following to add an action to modify a detail segment:

```
Dim action As UpdaterClient.IUADsegUpdate
Set action = transaction.AddUpdaterAction(UActionType_DSegUpdate)
```

You can then modify the properties of action variable to change the detail segment.

Properties (Listed)

The following table describes the properties of the IUADsegUpdate interface.

Table 22: IUADsegUpdate Properties

Property	Description
SaveMemo	Whether or not the memo should be saved.

Property	Description
ActionType	Type of action.
Action_IIDStr	String representaion of GUID that represents the action.
ProgID	Name of module that submitted the action.
Old_Seg_SK	SK for segment being modified.
Old _Emp_SK	SK of employee originally assigned to segment.
Old _Nom_Date	Original Nominal date of segment.
Old _Seg_Code_SK	SK for original segment code SK assigned to segment.
Old _Memo	Original memo for segment.
New_Seg_SK	SK for segment being modified.
New_Emp_SK	SK of employee SK being assigned to segment.
New_Nom_Date	New nominal date of segment.
New_Seg_Code_SK	SK of new segment code SK being assigned to segment.
New_Memo	New memo for segment.
Old_Start_Moment	Start moment for segment being modified.
Old_Stop_Moment	Stop moment for segment being modified.
New_Start_Moment	Start moment for segment being modified.
New_Stop_Moment	Stop moment for segment being modified.

Properties (Documented)

IUADsegUpdate::SaveMemo

This property indicates whether the memo specified in Memo property should be saved.

Settings/ Return values	Sets and returns a TBool value that specifies whether the memo specified in the Memo property should be saved to the database.
Remarks	The SaveMemo property is used to indicate to Updater whether it save the memo specified in the Memo property. This property applies only to actions that *modify* general and detail segments. This property allows you to not have to retrieve the memo for those actions if you are not going to change it. For example, when modifying either a general or a detail segment if you are not going to change the existing memo for the segment, you can set SaveMemo to false. That will save all the other changes you make for the segment

	but preserve the existing memo, if any. *For all other actions, this property is ignored.*

IUADsegUpdate::ActionType

This property indicates what kind of action this is.

Settings/ Return values	Read-only property of TUActionTypethat indicates what kind of action is to be performed.
Remarks	The value of ActionType is determined when the action is created by a call to IUpdaterTransaction.AddUpdaterAction.

IUADsegUpdate::Action_IIDStr

This is a read-only property. It returns the interface ID of the interface in registry string format.

IUADsegUpdate::ProgID

This property indicates the program or module that submitted this action.

Settings/ Return values	Returns a String property that indicates the name of the program or module that submitted this action.
Remarks	Read-only property that returns a String indicating the name of the program or module that submitted this action.

IUADsegUpdate::Old_Seg_SK

This property indicates the SK for the segment being modified.

Settings/ Return values	Sets and returns a Double value that indicates the SK for segment being modified.
Remarks	Set this property to the value of the SK of the segment you wish to modify.

IUADsegUpdate::Old_Emp_SK

This property indicates the SK of the employee originally assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the employee originally assigned to this segment.
Remarks	Set this property to the SK of the employee (EMP_SK from the EMP table) that was originally assigned to this segment. To change the employee assigned to this segment, change New_Emp_SK to the SK of the new employee.

IUADsegUpdate::Old_Nom_Date

This property indicates the original nominal date for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the original nominal date of the segment.
Remarks	Set this property to the original nominal date of the segment.

IUADsegUpdate::Old_Seg_Code_SK

This property indicates the SK of the original segment code assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the segment code originally assigned to the segment.
Remarks	You must set this property to the SK of segment code (SEG_CODE_SK from the SEG_CODE) originally assigned to the segment.

IUADsegUpdate::Old_Memo

This property indicates the original memo for the segment.

Settings/ Return values	Sets and returns a String value that indicates the original memo for the segment.
Remarks	Set this value to the original memo for the segment that you retrieve from the database. You must do this if you wish to change the memo for the segment, which can be done by setting the New_Memo property to the value you wish to change the memo to. If you wish to preserve the old memo, set SaveMemo to Bool_False and leave this property blank.

IUADsegUpdate::New_Seg_SK

This property indicates the SK for the segment being modified.

Settings/ Return values	Sets and returns a Double value that indicates the SK for segment being modified.
Remarks	For actions that modify a segment, this property is linked to the Old_Seg_SK property, which is used to select the segment being modified. Do not change this value. Use the Old_Seg_SK property instead.

IUADsegUpdate::New_Emp_SK

This property indicates the SK of the employee being assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the employee being assigned to this segment.
Remarks	Set this property to the SK of the employee (EMP_SK from the EMP table) to whom you are assigning this segment. If you are changing the employee assigned to this segment, change it from the value that was retrieved from the database, which must be set in Old_Emp_SK.

IUADsegUpdate::New_Nom_Date

This property indicates the nominal date for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the nominal date that the segment is being changed to.
Remarks	Set this property to the nominal date to which you wish to change this segment.

IUADsegUpdate::New_Seg_Code_SK

This property indicates the SK of the segment code assigned to this segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the segment code that the segment is being changed to.
Remarks	Set this property to the SK of the segment code (SEG_CODE_SK from the SEG_CODE) that you wish to change this segment to.

IUADsegUpdate::New_Memo

This property indicates the memo for this segment.

Settings/ Return values	Sets and returns a String value that indicates the memo for this segment.
Remarks	Set this property to the new value you wish to have for the memo of the segment. If you wish to preserve the old memo of the segment, you can leave this property blank if you also set SaveMemo to Bool_False.

IUADsegUpdate::Old_Start_Moment

This property indicates the original start moment of the segment.

Settings/ Return values	Sets and returns a Long value that indicates the original start moment for the segment.
Remarks	Set the Old_Start_Moment to the original start moment of the segment you are modifying.

IUADsegUpdate::Old_Stop_Moment

This property indicates the original stop moment of the segment.

Settings/ Return values	Sets and returns a Long value that indicates the original stop moment for the segment.
Remarks	Set the Old_Stop_Moment to the original stop moment of the segment you are modifying.

IUADsegUpdate::New_Start_Moment

This property indicates the new start moment for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the new start moment for the segment.

Remarks	Set the New_Start_Moment to the new start moment you want to assign to the segment.

IUADsegUpdate::New_Stop_Moment

This property indicates the new stop moment for the segment.

Settings/ Return values	Sets and returns a Long value that indicates the new stop moment for the segment.
Remarks	Set the New_Stop_Moment to the new stop moment you want to assign to the segment.

IUADsegDelete

The IUADsegDelete interface enables you to delete a detail segment.

Implementation

The IUADsegDelete interface is implemented by the UADsegDeleteCoClass defined in the UpdaterClient type library. However, you cannot create an instance of a UADsegDeleteCoClass directly. You must choose to add an action to a transaction and choose the appropriate TUActionType. For example, you can do the following to add an action to modify a detail segment:

```
Dim action As UpdaterClient.IUADsegDelete
Set action = transaction.AddUpdaterAction(UActionType_DSegDelete)
```

You can then modify the properties of action variable to the delete the detail segment

Properties (Listed)

The following table describes the properties of the IUADsegDelete interface.

Table 23: IUADsegDelete Properties

Property	Description
SaveMemo	This property is not used for this action.
ActionType	Type of action.
Action_IIDStr	String representaion of GUID that represents the action.
ProgID	Name of module that submitted the action.
Old_Seg_SK	SK for segment being deleted.
Old _Emp_SK	SK of employee assigned to segment being deleted.
Old _Nom_Date	Nominal date of segment being deleted.
Old _Seg_Code_SK	SK for segment code SK assigned to segment being deleted.
Old _Memo	Memo of segment being deleted.

Property	Description
Old_Start_Moment	Start moment for segment being deleted.
Old_Stop_Moment	Stop moment for segment being deleted.

Properties (Documented)

IUADsegDelete::SaveMemo

This property indicates whether the memo specified in Memo property should be saved.

Settings/ Return values	NA
Remarks	This property is not used for this action.

IUADsegDelete::ActionType

This property indicates what kind of action this is.

Settings/ Return values	Read-only property of TUActionTypethat indicates what kind of action is to be performed.
Remarks	The value of ActionType is determined when the action is created by a call to IUpdaterTransaction.AddUpdaterAction.

IUADsegDelete::Action_IIDStr

This is a read-only property. It returns the interface ID of the interface in registry string format.

IUADsegDelete::ProgID

This property indicates the program or module that submitted this action.

Settings/ Return values	Returns a String property that indicates the name of the program or module that submitted this action.
Remarks	Read-only property that returns a String indicating the name of the program or module that submitted this action.

IUADsegDelete::Old_Seg_SK

This property indicates the SK for the segment being deleted.

Settings/ Return values	Sets and returns a Double value that indicates the SK for segment being deleted.
Remarks	Set this property to the value of the SK of the segment you wish to delete.

IUADsegDelete::Old_Emp_SK

This property indicates the SK of the employee assigned to the segment being deleted.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the employee assigned to the segment being deleted.
Remarks	Set this property to the SK of the employee (EMP_SK from the EMP table) assigned to the segment being deleted.

IUADsegDelete::Old_Nom_Date

This property indicates the nominal date of the segment being deleted.

Settings/ Return values	Sets and returns a Long value that indicates the nominal date of the segment being deleted.
Remarks	Set this property to the nominal date of the segment being deleted.

IUADsegDelete::Old_Seg_Code_SK

This property indicates the SK of the segment code assigned to the segment.

Settings/ Return values	Sets and returns a Double value that indicates the SK for the segment code assigned to the segment.
Remarks	You must set this property to the SK of segment code (SEG_CODE_SK from the SEG_CODE) assigned to the segment being deleted.

IUADsegDelete::Old_Memo

This property indicates the memo for the segment being deleted.

Settings/ Return values	Sets and returns a String value that indicates the memo for the segment.
Remarks	Set this value to the original memo for the segment that you retrieve from the database. You must do this to delete the segment.

IUADsegDelete::Old_Start_Moment

This property indicates the start moment of the segment.

Settings/ Return values	Sets and returns a Long value that indicates the start moment of the segment.
Remarks	Set the Old_Start_Moment to the start moment of the segment you are deleting..

IUADsegDelete::Old_Stop_Moment

This property indicates the stop moment of the segment.

Settings/ Return values	Sets and returns a Long value that indicates the stop moment of the segment.
Remarks	Set the Old_Stop_Moment to the stop moment of the segment you are deleting.

IUpdaterViolation

The IUpdaterViolation interface enables you to examine the strict-type and warning-type violations that are returned when a transaction fails.

Implementation

The IUpdaterViolation interface is implemented by the UpdaterViolationCoClass defined in the UpdaterClient type library. You cannot create an UpdaterViolationCoClass to get a reference to an IUpdaterViolation interface. The IUpdaterViolation interface can be retrieved from either the StrictViolations or WarningVioloations property of the transaction:

```
Dim violation1 As UpdaterClient.IUpdaterViolation
Dim violation2 As UpdaterClient.IUpdaterViolation

Set violation1 = transaction.StrictViolations(0)
Set violation2 = transaction.WarningViolations(0)
```

You can then use the violation variable to determine which rules were violated, and take action to rectify the problems with your transaction.

Properties (Listed)

The following table describes the properties of the IUpdaterViolation interface.

Table 24: IUpdaterViolation Properties

Property	Description
Seg1Key	SK of the first segment in the violation.
Seg1Detail	Whether the first segment in violation is a detail segment.
Seg1EmployeeKey	SK of employee assigned to first segment in violation.
Seg1NominalDate	Nominal date of first segment in violation.
Seg2Key	SK of the second segment in the violation.
Seg2Detail	Whether the second segment in violation is a detail segment.
Seg2EmployeeKey	SK of employee assigned to second segment in violation.
Seg2NominalDate	Nominal date of first segment in violation.
ErrorCode	Error code for violation.
ErrorMessage	Error message pertaining to the violation.
RuleProgID	ProgID of the COM object that determined the violation. This would be populated for Updater plug-in rules and would be blank for internal rules.
RuleKey	SK of user-defined rule violated.
ResourceID	Resource ID for violation.

Properties (Documented)

IUpdaterViolation::Seg1Key

This property indicates the SK of the first segment in the violation.

Settings/ Return values	Read-only Double property that returns the SK of the first segment in the violation.
Remarks	For all rules that involve more than one segment, Seg1Key indicates the SK for the first segment in the violation. Otherwise it indicates the SK of the only segment in the violation.

IUpdaterViolation::Seg1Detail

This property indicates whether the first segment in the violation is a detail segment.

Settings/ Return values	Read-only TBool property that indicates whether the first segment in the violation is a detail segment.
Remarks	Indicates whether the segment represented by Seg1Key is a detail segment. If Bool_True then the segment is a detail segment, otherwise it is a general segment.

IUpdaterViolation::Seg1EmployeeKey

This property indicates the SK of the employee assigned to the first segment in the violation.

Settings/ Return values	Read-only Double property that indicates the SK of the employee assigned to first segment in the violation.
Remarks	Seg1EmployeeKey is the SK of the employee assigned to the first segment in the violation.

IUpdaterViolation::Seg1NominalDate

This property indicates the nominal date of the first segment in the violation.

Settings/ Return values	Read-only Long property that indicates the nominal date of the first segment in the violation.
Remarks	Seg1NominalDate is the nominal date of the first segment in the violation.

IUpdaterViolation::Seg2Key

This property indicates the SK of the second segment in the violation.

Settings/ Return values	Read-only Double property that returns the SK of the second segment in the violation.
Remarks	For all rules that involve more than one segment, Seg2Key indicates the SK for the second segment in the violation.

IUpdaterViolation::Seg2Detail

This property indicates whether the second segment in the violation is a detail segment.

Settings/ Return values	Read-only property that returns a TBool value indicating whether the second segment in the violation is a detail segment.
Remarks	Indicates whether the segment represented by Seg2Key is a detail segment. If the Bool_True then the segment is a detail segment, otherwise it is a general segment.

IUpdaterViolation::Seg2EmployeeKey

This property Indicates the SK of the employee assigned to the second segment in the violation.

Settings/ Return values	Read-only Double property that indicates the SK of the employee assigned to second segment in the violation.
Remarks	Seg2EmployeeKey is the SK of the employee assigned to the second segment in the violation.

IUpdaterViolation::Seg2NominalDate

This property indicates the nominal date of the second segment in the violation.

Settings/ Return values	Read-only Long property that returns the nominal date of the second segment in the violation.
Remarks	Seg2NominalDate is the nominal date of the second segment in the violation.

IUpdaterViolation::ErrorCode

This property indicates the error code of the violation.

Settings/ Return values	Read-only Long property that returns the error code of the violation.
Remarks	ErrorCode indicates the type of rule that was violated. For more information on the error codes, see Table 14.

IUpdaterViolation::ErrorMessage

This property indicates the message pertaining to the violation.

Settings/ Return values	Read-only BSTR property that returns information about the violation.
Remarks	ErrorMessage provides descriptive text about the violation.

IUpdaterViolation::RuleProgID

This property indicates the ProgID of the Updater plug-in rule COM object that determined the violation.

Settings/ Return values	Read-only BSTR property that returns RuleProgID.

Remarks	RuleProgID provides the ProgId of the Updater plug-in rule that determined the violation. It would be blank for internal rules.

IUpdaterViolation::RuleKey

This property indicates the SK of the user-defined rule that was violated.

Settings/ Return values	Read-only Double property that indicates the user-defined rule that was violated.
Remarks	RuleKey is the SK of a user-defined rule of either the “must be contained in” or the “must not overlap” rule types that was violated.

IUpdaterViolation::ResourceID

This property indicates the resourceID of the violation.

Settings/ Return values	Read-only Long property that returns the resource ID of the violation.
Remarks	ResourceID is the resource ID of the rule violated, which is used in displaying the error to the user in the Aspect Workforce application.

IUpdaterError

The IUpdaterError interface enables you to examine the errors that are returned when a transaction fails.

Implementation

The IUpdaterError interface is implemented by the UpdaterErrorCoClass defined in the UpdaterClient type library. You cannot create an UpdaterErrorCoClass to get a reference to an IUpdaterError interface. The IUpdaterError interface can be retrieved from the Errors property of the transaction.

```
Dim error As UpdaterClient.IUpdaterError
Set error = transaction.Errors(0)
```

You can then use the error variable to determine the errors encountered by Updater in processing the transaction, and take action to rectify the problems with your transaction.

Properties (Listed)

The following table describes the properties of the IUpdaterError interface.

Table 25: IUpdaterError Properties

Property	Description
ErrorCode	Error code of Updater error.
ErrorMessage	Error message pertaining to the violation.
RuleProgID	ProgID of the COM object that determined the violation. This would be populated for Updater plug-in rules and would be blank for internal rules.

Property	Description
ResourceID	Resource ID for violation.

Properties (Documented)

Following are descriptions of all properties of the IUpdaterError interface.

IUpdaterViolation::ErrorCode

This property indicates the error code.

Settings/ Return values	Read-only Long property that returns the error code of the error.
Remarks	*ErrorCode*: An error encountered by Updater while processing the transaction. ErrorCode indicates the type of rule that was violated. For more information on the error codes, see Table 14.

IUpdaterViolation::ErrorMessage

This property indicates the message pertaining to the violation.

Settings/ Return values	Read-only BSTR property that returns information about the violation.
Remarks	ErrorMessage provides descriptive text about the violation.

IUpdaterViolation::RuleProgID

This property indicates the ProgID of the Updater plug-in rule COM object that determined the violation.

Settings/ Return values	Read-only BSTR property that returns RuleProgID.
Remarks	RuleProgID provides the ProgId of the Updater plug-in rule that determined the violation. It would be blank for internal rules.

IUpdaterViolation::ResourceID

This property indicates the resourceID of the error.

Settings/ Return values	Read-only Long property that returns the resource ID of the error.
Remarks	ResourceID is the resource ID of the error, which is used in displaying the error to the user in the Aspect Workforce application.

IUpdaterSuperstateSummaryFilter

The IUpdaterSuperstateSummaryFilter interface enables you to specify the list of superstates for which superstate hours should be calculated. This section lists the methods and properties of the

IUpdaterSuperstateSummaryFilter interface along with brief descriptions of each. It also discusses how to implement the interface.

Implementation

The IUpdaterSuperstateSummaryFilter interface is implemented by the UpdaterTransactionCoClass defined in the UpdaterClient type library. To create an UpdaterTransactionCoClass and get a reference to an IUpdaterSuperstateSummaryFilter interface, you can do the following:

```
Dim filter As UpdaterClient.IUpdaterSuperstateSummaryFilter
Set filter = New UpdaterClient.UpdaterTransactionCoClass
```

You can then use the filter variable to access the properties and methods of the IUpdaterSuperstateSummaryFilter interface.

Methods (Listed)

The following table lists and briefly describes the methods of the IUpdaterSuperstateSummaryFilter interface.

Table 26: IUpdaterSuperstateSummaryFilter Methods

Method	Description
GetIncludeAll	Returns true if the list of superstate keys is empty, which means that all superstates should be included and false otherwise.
Clear	Clears the list of superstate keys.
AddSuperstate	Adds a superstate to the key list.
GetSuperstate	Returns the superstate key at a specified index.
Count	Returns the number of superstate keys in the list.

Methods (Documented)

This section documents the methods of the IUpdaterSuperstateSummaryFilter interface.

IUpdaterSuperstateSummaryFilter:: GetIncludeAll ()

This method returns true if the list of superstate keys is empty, which means that all superstates should be included and false otherwise.

Parameters	None.
Return values	Returns a boolean.
Remarks	The default value is true.

IUpdaterSuperstateSummaryFilter:: Clear ()

This method clears the list of superstate keys.

Parameters	None.
Return values	None.
Remarks	Immediately after calling this method all superstates will be included.

IUpdaterSuperstateSummaryFilter:: AddSuperstate (superstateSK As Double)

This method adds a superstate to the key list.

Parameters	superstateSK: A double value that holds a superstate SK value to add to the key list.
Return values	None.
Remarks	None.

IUpdaterSuperstateSummaryFilter:: Count ()

This method returns the number of superstate keys in the list.

Parameters	None.
Return values	Returns an integer value that specifies the number of superstate keys in the list.
Remarks	None.

IUpdaterSuperstateSummaryFilter:: GetSuperstate (Idx As integer)

This method returns the superstate key at the specified index.

Parameters	*Idx*: An integer value that specifies an index into the list of superstate keys.
Return values	A double value that holds the superstate key at the specified index.
Remarks	The index into the superstate key list is based.

TBool

Values from the TBool enumeration are used to indicate true or false attributes in Updater's COM interfaces.

```
enum _TBool
       {
       Bool_False = 0,
       Bool_True = 1

} TBool;
```

Table 27: TBool Elements

Element	Description
Bool_True	Indicates a True value.
Bool_False	Indicates a False value.

TUActionType

Values from the TUActionType determine the type of the Updater action.

```
enum _TUActionType
{
       UActionType_None = 0,
UActionType_GSegInsert = 1,
UActionType_DSegInsert = 2,
UActionType_GSegUpdate = 3,
UActionType_DSegUpdate = 4,
UActionType_GSegDelete = 5,
UActionType_DSegDelete = 6,
UActionType_Summarize = 7
} TUActionType;
```

Table 28: TUActionType Elements

Element	Description
UActionType_None	Indicates a no-op action.
UActionType_GSegInsert	Indicates an action to insert a general segment.
UActionType_DSegInsert	Indicates an action to insert a detail segment.
UActionType_GSegUpdate	Indicates an action to update a general segment.
UActionType_DSegUpdate	Indicates an action to update a detail segment.
UActionType_GSegDelete	Indicates an action to delete a general segment.
UActionType_DSegDelete	Indicates an action to delete a detail segment.
UActionType_Summarize	Indicates an action to summarize the daily segment summary.

The values from the TUActionType enumeration are used to create the different actions in a transaction. The individual actions store a value from the enumeration corresponding to their action in their ActionType property. The type of the action cannot be changed once it has been created.

TUPriority

Values from the TUPriority enumeration indicate the priority of the transaction.

```
enum _TUActionType
enum _TUPriority
{
       UPriority_Low = 100,
UPriority_Normal = 200,
UPriority_High = 300
} TUPriority;
```

Table 29: TUPriority Elements

Element	Description
UPriority_Low	Indicates a low transaction priority.
UPriority_Normal	Indicates a normal transaction priority.
UPriority_High	Indicates a high transaction priority.

The Priority property of a transaction can be set to one of the values of the TUPriority enumeration to indicate its priority in the Updater system. Typically this should be set to UPriority_Normal. However, certain tasks that require immediate evaluation by Updater may be submitted with a priority of UPriority_High. Other batch entry jobs can be submitted with a priority of UPriority_Low.

TUStatus

Values from the TUStatus enumeration indicate the status of the transaction in the Updater system.

```
enum _TUStatus
{
       UStatus_Inactive = 0,
UStatus_Executing_On_Client = 1,
       UStatus_Queued_On_Server = 2,
       UStatus_Executing_On_Server = 3,
       UStatus_Succeeded = 4,
       UStatus_Failed = 5,
       UStatus_Canceled = 6,
       UStatus_Unknown = 7
} TUStatus;
```

Table 30: TUStatus Elements

Element	Description
UStatus_Inactive	Indicates that the transaction is inactive. The transaction is in this state before it is submitted to Updater.
UStatus_Executing_On_Client	Indicates that the transaction is being processed by the Updater client.
UStatus_Queued_On_Server	Indicates that the transaction has been submitted to the Updater server for evaluation. At this stage, there are other transactions that are being processed and this transaction is in queue to be processed.

Element	Description
UStatus_Executing_On_Server	Indicates that the transaction is currently being evaluated by the Updater server.
UStatus_Succeeded	Indicates that the transaction has been completely evaluated, no errors or strict-enforcement type violations were found, and also that the transaction was successfully committed to the database.
UStatus_Failed	Indicates that the transaction failed for some reason. Look in the StrictViolations and Errors property of the transaction to determine the cause for failure.
UStatus_Canceled	Indicates that the transaction was canceled.
UStatus_Unknown	Indicates that the Updater client could not determine the status of the transaction. This is rare.

The Status property of the transaction can have one of the values of the TUStatus enumeration. Use the Status property of the transaction at any time after creating a transaction to determine its current status. You can also use it after the transaction has completed processing to determine whether the transaction succeeded or failed.

Other Notes

This section provides notes on the Updater type libraries, programming with other languages, and using the custom interfaces.

About the Updater Type Libraries

There are two type libraries in the Updater API. You can use any standard type library viewer to browse the libraries.

UpdaterCommonUp

File Name: UpCommon.dll

The UpdaterCommon type library contains several types that are used by both the Updater client as well as the Updater server. Some of these types are exposed required in the use of the Updater API whereas others are only for use within the Updater system.

UpdaterClient

FileName: UpClient.dll

The UpdaterClient type library contains types that are used by the Updater client to allow applications to communicate with it. It exposes several interfaces and objects that application developers use to submit segment changes to Updater.

Programming with Other Languages

The Updater API can be accessed using any programming language and development environment that supports COM programming. Note that all the interfaces in the Updater API are dual interfaces, which allows them to be accessed via both an automation interface as well as a custom interface.

To use the Updater API from a development environment other than Visual Basic, refer to the documentation for the environment.

Using the Custom Interfaces

To use the custom interfaces (vtable based calling) with a development environment, such as C++ or Borland Delphi, you will need to import the type library definitions into your development environment. That will create several files that you can include in your application. Then, you can use the interfaces provided in the Updater API in the same way as described in this document with the added advantage of increased execution speed of your application. Refer to the documentation accompanying your development environment to generate the files that you can include in your application.

Visual Basic Reference

The wrapper files written in Visual Basic version 6 have been removed from this SDK. We strongly encourage you to use the functions included in Alvaria Web Services to submit official schedule changes.

Locale Data Format

This section defines the content of IUpdaterTransaction::ClientLocale property. The content consists of an XML string formatted in compliance with the “Locale Data Markup Language” as defined at http://www.openi18n.org/spec/ldml/1.0/ldml-spec.htm. This specification has optional fields as well as accommodation for Windows-specific information. The following example illustrates the minimal content of this XML string.

Sample Code

```
<ldml>
	<identity>
		<version number="1.0" />
		<generation date="2005-04-21" />
		<language type="en" />
		<territory type="US" />
	</identity>
	<dates>
	<calendars>
	<calendar type="gregorian">
	<monthNames>
		<month type="1">January</month>
		<month type="2">February</month>
		<month type="3">March</month>
		<month type="4">April</month>
		<month type="5">May</month>
		<month type="6">June</month>
		<month type="7">July</month>
		<month type="8">August</month>
		<month type="9">September</month>
		<month type="10">October</month>
		<month type="11">November</month>
		<month type="12">December</month>
	</monthNames>
	<monthAbbr>
		<month type="1">Jan</month>
		<month type="2">Feb</month>
		<month type="3">Mar</month>
```

```
        <month type="4">Apr</month>
        <month type="5">May</month>
        <month type="6">Jun</month>
        <month type="7">Jul</month>
        <month type="8">Aug</month>
        <month type="9">Sep</month>
        <month type="10">Oct</month>
        <month type="11">Nov</month>
        <month type="12">Dec</month>
</monthAbbr>
<dayNames>
        <day type="sun">Sunday</day>
        <day type="mon">Monday</day>
        <day type="tue">Tuesday</day>
        <day type="wed">Wednesday</day>
        <day type="thu">Thursday</day>
        <day type="fri">Friday</day>
        <day type="sat">Saturday</day>
</dayNames>
<dayAbbr>
        <day type="sun">Sun</day>
        <day type="mon">Mon</day>
        <day type="tue">Tue</day>
        <day type="wed">Wed</day>
        <day type="thu">Thu</day>
        <day type="fri">Fri</day>
        <day type="sat">Sat</day>
</dayAbbr>
<am>AM</am>
<pm>PM</pm>
<eras>
<eraName>
        <era type="0" />
        <era type="0" />
        <era type="0" />
        <era type="0" />
        <era type="0" />
        <era type="0" />
        <era type="0" />
</eras>
<dateFormats>
        <default type="short" />
        <dateFormatLength type="long">
        <dateFormat>
             <pattern>dddd, MMMM dd, yyyy</pattern>
        </dateFormat>
        </dateFormatLength>
        <dateFormatLength type="short">
        <dateFormat>
             <pattern>M/d/yyyy</pattern>
        </dateFormat>
        </dateFormatLength>
</dateFormats>
<timeFormats>
        <default type="short" />
        <timeFormatLength type="long">
        <timeFormat>
             <pattern>h:mm:ss AMPM</pattern>
        </timeFormat>
        </timeFormatLength>
        <timeFormatLength type="short">
        <timeFormat>
             <pattern>h:mm MPM</pattern>
```

```
        </timeFormat>
              </timeFormatLength>
        </timeFormats>
        </calendar>
        </calendars>
        </dates>
        <numbers>
        <symbols>
        <decimal>.</decimal>
        <group>,</group>
        <list>,</list>
        </symbols>
        <currencyFormats>
              <currencyFormatLength type="long">
              <currencyFormat>
              <pattern>$#,##0.00;($#,##0.00)</pattern>
        </currencyFormat>
              </currencyFormatLength>
        </currencyFormats>
        <currencies>
              <currency type="USD">
                    <symbol>$</symbol>
              </currency>
        </currencies>
        </numbers>
        <special xmlns:alvaria=http://www.alvaria.com
              < alvaria:win32_locale>
              < alvaria:LCID>1033</ alvaria:LCID>
              < alvaria:PrimaryLangID>9</ alvaria:PrimaryLangID>
              < alvaria:SubLanID>1</ alvaria:SubLanID>
              < alvaria:FarEastFlag>0</ alvaria:FarEastFlag>
              < alvaria:MiddleEastFlag>0</ alvaria:MiddleEastFlag>
              < alvaria:currencyFormat LCTYPE="LOCALE_ICURRENCY">0<
              alvaria:
              currencyFormat>
              < alvaria:negCurrFormat LCTYPE="LOCALE_INEGCURR">0<
              alvaria:
              negCurrFormat>
              < alvaria:currencyDecimals
              LCTYPE="LOCALE_ICURRDIGITS">2<
              alvaria:currencyDecimals>   < alvaria:DateSeparator
              LCTYPE="LOCALE_SDATE">/</ alvaria:Date
              Separator>
              < alvaria:TimeSeparator LCTYPE="LOCALE_STIME">:<
              alvaria:TimeSeparator>
              </ alvaria:win32_locale>
        </special>
</ldml>
```

Updater Plug-In Rule Developer Guide

This guide specifies the necessary information necessary to create, implement, and install an Updater plug-in rule in Aspect Workforce. The guide defines in detail how to code theses functions, create the DLL, and configure and install the plug-in rule.

Related Software Files

The SDK includes a sample program and related files that demonstrate how to program an Updater plug-in rule. After you install the SDK, these files can be found at the following default path:

```
C:\Program Files (x86)\Alvaria\Workforce 25.0
SDK\API_Related_Files\Updater_Plug-In_Rule_Demo
```

The sample program, written in Visual Basic .Net, supports Windows Server 2022 and Windows Server 2025.

Prerequisites

Table 1 shows the requirements for creating Updater plug-in rules.

Table 1: Aspect Workforce Requirements for Updater Plug-In Rules

Versions	Aspect Workforce 7.5 or later
Enhancement Packages	None
Database	Aspect Workforce database must be installed
Other Software	• Updater service must be installed • Aspect Workforce client must be installed

Audience

This SDK is intended for developers who are extending the capabilities of the Updater Server to support additional rule checking. This document assumes you are skilled with the following:

- A programming environment that supports COM/COM+
- Aspect Workforce Updater API
- Aspect Workforce

List of Features

This guide enables you to implement the following features:

- The capability of developing a plug-in rule DLL.
- The capability of integrating the plug-in rule DLL into the Aspect Workforce system.
- The capability of controlling the order that plug-in rules are invoked.

Organization of this Guide

The sections of this document are organized as follows:

- About Updater Plug-In Rules - presents a high-level overview of the purpose and implementation methodology of plug-in rules.
- Architecture - describes how plug-in rules fit into the Aspect Workforce architecture.
- Implementation Steps - describes the steps necessary to implement a plug-in rule.
- Deployment Steps - describes the steps necessary to deploy a plug-in rule.
- API - describes the API that must be presented by the plug-in rule DLL.
- Interaction Diagram - depicts the function call order for invoking a plug-in rule.
- Error Messages - describes how to convey error message from the plug-in rule back to Updater.
- Type Library - defines the name and location of the type library that is necessary to create the plug-in rule COM DLL.
- Performance Guidelines - highlights performance considerations pertinent to developing plug-in rules.
- Installing and Configuring Plug-In Rules - describes how to install and configure plug-in rules.
- Sample Program - provides information about the sample program.

About Updater Plug-In Rules

An Updater plug-in rule is a Component Object Model (COM) component that can be developed by a third-party, such as a system integrator, and integrated with the Aspect Workforce Updater service. Typically, Updater plug-in rules contain business logic that is not part of the standard functioning and validating processes of Updater, but that might be needed to meet the special needs of a user. For example: the Minimum Shift Break Rule, which ensures that a specified minimum period elapses between the end of an employee shift and the beginning of the next shift.

A plug-in rule is manifested via a DLL with a COM interface. This COM interface is invoked by Updater as part of its rule checking. The COM interface must provide the following functions:

```
GetConfigurationInfo(…)

Validate(…)
```

Architecture

It is important to understand where the plug-in rule DLL that you create fits into the Aspect Workforce system. This section discusses, from two perspectives, where your DLL fits into the Aspect Workforce architecture. The first perspective is the architecture supporting the validation of the plug-in rules. The

second perspective is the architecture that supports the entry of plug-in rule configuration data. A discussion of each follows.

Rule Validation

Figure -1 depicts the architecture for invoking the plug-in rule for the purposes of validating that the segment changes pass or fail.

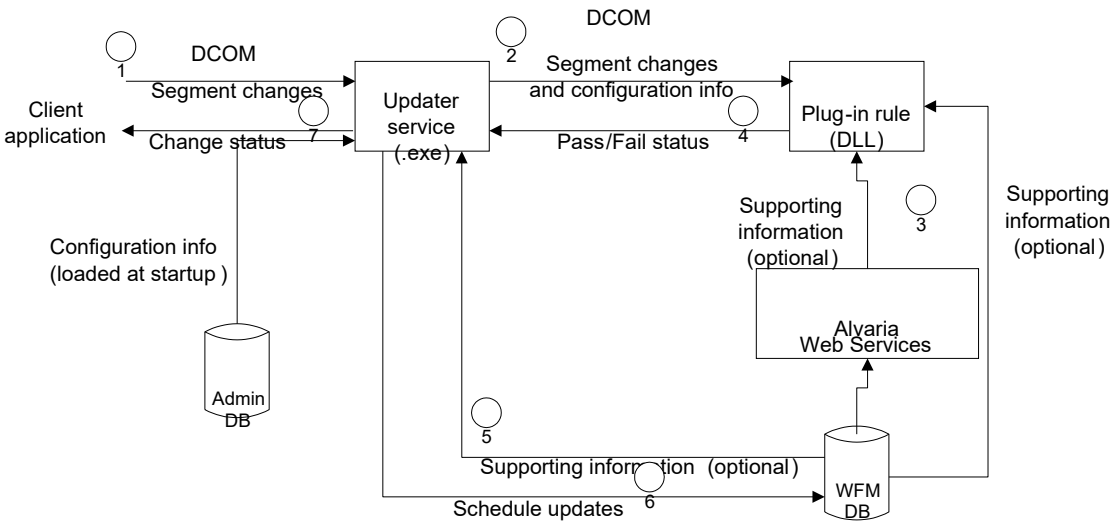


Figure -1: Architecture for Rule Validation

It is important to understand where the plug-in rule DLL that you write fits into the Aspect Workforce system. The label in Figure -1 titled “Plug-In Rule (DLL)” depicts where your DLL fits into the system. To clarify how and when this is invoked, a discussion follows of the processing path, based on the numbers in the figure.

1. The client, typically the Rich Client, invokes the “Updater Service (exe)” with a list of segment changes.

2. The “Updater Service (exe)” invokes the “Plug-In Rule (DLL)” Validate function passing it the list of segment changes and the configuration information.

3. The “Plug-In Rule (DLL)” optionally can obtain information from the “WFM Web services” and/or directly from the “WFM DB”. This information would be used to determine if the segment changes pass/fail.

4. After the “Plug-In Rule (DLL)” determines which segments pass/fail, the pass/fail status is sent back to the “Updater Service (exe)”.

5. The “Updater Service (exe)” optionally can obtain information from the “WFM DB” for the purposes of discerning pass/fail status based on Updater’s internal rules. Note that when the Updater internal rules are applied is configurable. This example shows the internal rules being applied after the plug-in rules. However, they could be applied before or after any particular plug-in rule. See Invoking the Plug-In Rule for additional information on the ordering of plug-in rules.

6. Updater makes the changes to the “WFM DB” for the segment changes that passed both the plug-in rules and the internal rules with the following restriction. All segments in a transaction (for example, a segment change request, such as in step 1 in Figure -1) must pass both the internal and plug-in rules in order for Updater to make the changes in the “WFM DB”. In other words, it is a all or nothing proposition.

7. The status of the changes is passed back to the client.

8. Notice in Figure -2 that the “Configuration Info” is loaded at startup of the “Updater Service (exe)”. It is important to be aware of this because changes to the configuration information will not take affect until the next time “Updater Service (exe)” is started. See Plug-In Rule Configuration Information for details about the creation and persistence of configuration information.

Plug-In Rule Configuration Information

Figure -2 depicts the architecture for entering the Updater plug-in rule configuration information.

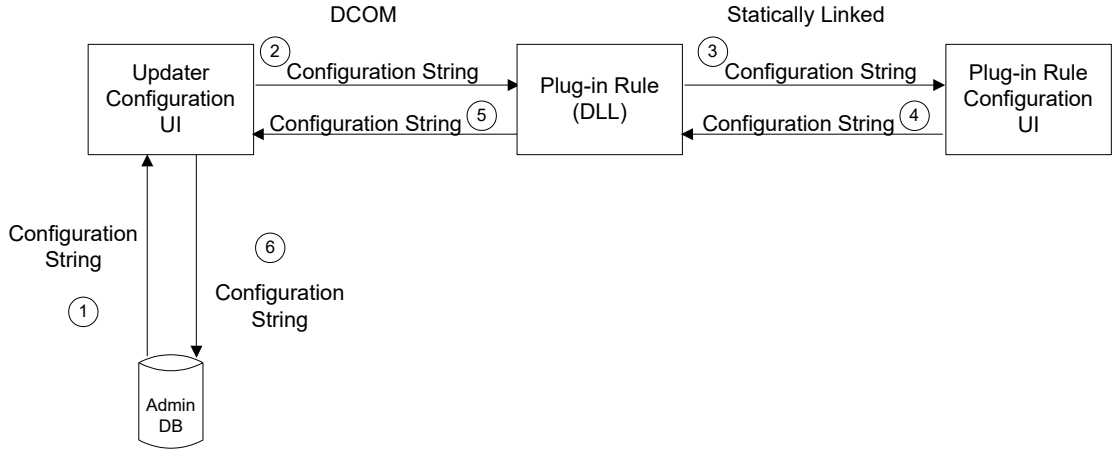


Figure -2: Architecture for Entering Updater Plug-In Rule Configuration Information

It is important to understand the difference between “Plug-In Rule Configuration Information” and “Updater Configuration Information”. “Plug-In Rule Configuration Information” refers to the information that the plug-in rule must have to operate. For example, if a plug-in rule enforces that a segment cannot be longer than X minutes, then the value of X would be plug-in rule configuration information. On the other hand, “Updater Configuration Information” refers to the information that is necessary to incorporate the “Plug-In Rule (DLL)” into the Aspect Workforce system. For example, the order of invoking plug-in rules would be defined by the Updater configuration information.

The steps for creating and persisting plug-in rule configuration information, based on the numbers in the figure, are as follows:

1. The “Updater Configuration UI” reads the configuration information from the Admin DB.

2. The “Updater Configuration UI” invokes the “Plug-In Rule (DLL)” and passes the configuration information.

3. The “Plug-In Rule (DLL)” invokes the “Plug-In Rule Configuration UI” and passes the configuration information.

4. “Plug-In Rule Configuration UI” presents a UI for editing the configuration string and passes the edited string back to the “Plug-In Rule (DLL)”.

5. The “Plug-In Rule (DLL)” passes the configuration string back to the “Updater Configuration UI”.

6. The “Updater Configuration UI” stores the string back in the Admin DB.

Installing and Configuring Plug-In Rules explains how to invoke the Plug-In Rule Configuration UI.

Workforce 8.2 and later and 64-bit

In Workforce 8.2 and later, much of the code executes in the 64-bit system. This includes the Updater service that will use your pluggable rule. In earlier versions, all programs ran as 32-bit applications.

For version 8.2 and later, your DLL must be built and installed so that it can run as both 32-bit and 64-bit. The Updater service will need to execute the DLL as a 64-bit DLL when it uses your DLL to validate the the Updater transaction. Service Installer will execute your DLL as a 32-bit DLL when you configure the rule. If your rule is built using C++, you will most likely need to build both 32-bit and a 64-bit versions, then install and register both. If your rule is built using .Net you may build the rule so that it executes under AnyCPU, which will work in both 32-bit and 64-bit.

Implementation Steps

It is important to understand from a macroscopic perspective the steps necessary to implement a plug-in rule. The lists of steps follows:

1. Identify the business rules that should be enforced via the plug-in rule.

2. Identify the data necessary to check the rule. Possible sources of data include the Aspect Workforce web services and accessing the Database directly. Accessing the web services is preferable to directly accessing the database; however, in some cases the web services may not provide the necessary data, and going directly to the DB would be required.

3. Design the algorithm that will enforce the business rules.

4. Determine what error strings should be returned for:

- Strict Violations
- Warning Violations
- AddError

Note: For strict and warning violations, error strings are returned via the IUpdaterViolation::ErrorMessage property. AddError messages are set via IUpdaterClient::AddError. For additional information, see Chapter 2, Updater API Developer Guide.

5. If the returned strings must support more than one language, design a system for selecting the string based on locale settings. See the IUpdaterTransaction::ClientLocale property description in Updater API Developer Guide, for information on how the locale information would be passed to the plug-in rule DLL.

6. Implement the plug-in rule DLL in accordance with the API described in API.

7. Build a test harness that will invoke the plug-in rule.

8. Validate, via the test harness, that the behavior and performance of the rule is acceptable. See Performance Guidelines for performance guidelines.

9. Deploy per the instructions in Deployment Steps.

Deployment Steps

The steps for deploying the plug-in rule follow:

1. Install the plug-in rule per the instructions in Installing the DLL.

2. Via the Updater Configuration UI, perform the setup steps such that Updater will invoke the plug-in rule DLL. See Configuring Updater for Plug-In Rulesfor instructions on how to use the Updater Configuration UI.

3. Via the Plug-In Rule Configuration UI, perform the steps to configure the plug-in rule. See Configuring Plug-In Rules for instructions.

API

To implement plug-in rules, it is necessary to understand the API that must be presented by the plug-in rule DLL. The plug-in rule DLL must present a DCOM API in the form of a class that inherits from the IWFMUpdaterRule interface. IWFMUpdaterRule defines the following functions:

```
GetConfigurationInfo(…)

Validate(…)
```

The plug-in rule class must define and thus override these functions. Details on each of these functions follows.

GetConfigurationInfo

The purpose of the GetconfigurationInfo function is to allow the entry of configuration information that is specific to the particular plug-in rule. Configuring Plug-In Rules discusses in detail the architecture supporting the entry of plug-in rule configuration data. When this function is invoked by the Updater Configuration UI (see Configuring Updater for Plug-In Rules), the function would typically present a UI that would support the entry of plug-in rule configuration data. The entered data would be persisted in the Admin DB as shown in Figure -2. The declaration for the GetConfigurationInfo function follows:

```
HRESULT _stdcall GetConfigurationInfo([in] BSTR DBAlias[in][out] BSTR
* ConfigString );
```

DBAlias

DBAlias is a string input value that identifies the Aspect Workforce DB that pertains to the configuration string. The DBAlias may or may not be used by the plug-in rule configuration UI. At a minimum, the plug-in rule configuration UI should probably display the DBAlias. In any case, it is important to understand how the DBAlias relates to uniquely identifying the configuration string in the Admin DB. From a conceptual perspective, the primary key for identifying the configuration string in the Admin DB consists of the following two values.

- Rule_ID – An ID assigned by the Updater Configuration UI.
- DB_ID – A database ID that is uniquely identified by the DBAlias.

Therefore, a particular Rule_ID could have multiple DB_IDs that reference different Aspect Workforce DBs. Since Updater can support multiple Aspect Workforce DBs currently, the above primary key

accommodates, for each Rule_ID, multiple Aspect Workforce DBs. Ultimately, this supports the entry of a unique configuration string for each Rule_ID and DB_ID combination.

It should be noted that the terms Rule_ID and DB_ID are used for conceptual purposes. The actual manifestation in the Admin DB may use different names for these entities as well as different schemas over time. However, the concept of a Rule_ID and DB_ID uniquely identifying a configuration string will always be valid.

ConfigString

ConfigString is both a string input and output value. The value going in is the value of the string that was persisted in the Admin DB. Typically, this value would be presented to the user in the Plug-In Rule Configuration UI, and the user could modify the string. The value returned from this function call would be persisted back to the Admin DB; and therefore, would replace the previous value.

The purpose of this string is to hold the configuration information that would be used by the Validate function (see Validate). For example, if the plug-in rule enforced a maximum allowable amount of time per shift, then the configuration string would contain the value for the maximum allowable time. The format of the string is purely at the plug-in rule designer's discretion. The string format could be name/value pairs, xml, or any other desired format.

Validate

The Validate function is called by Updater to determine if the segment changes violate the plug-in rules. The Validate function prototype is shown below. ConfigString contains the information that was entered via the Plug-In Rule Configuration UI described in Configuring Plug-In Rules. Transaction contains the segment changes as well as place holders for rule violation information. Therefore, for each segment change listed in Transaction, the validate function checks for rule violations. If rule violations are present, then the violation information would be written to Transaction. A detailed description of the two arguments follows.

```
HRESULT _stdcall Validate([in] BSTR ConfigString,
[in] IUpdaterTransaction * Transaction);
```

ConfigString

The string that was entered via the Plug-In Rule Configuration Tool. See GetConfigurationInfo for details about the content of this string.

Transaction

IUpdaterTransaction is the interface that is used by an Updater client and plug-in rules. It is documented in section 0. Updater API Developer Guide. The component class (i.e., implementation) is different for the two cases of the Updater client and plug-in rules. The component class for the plug-in rules is the TUpdaterRuleTransaction class. It is unique in that it has many functions that have been disabled. In other words, if these functions were invoked, then they would do nothing. It should be noted that invoking these functions does not harm the system in any way.

Disabled functions in the TUpdaterRuleTransaction class are:

- TUpdaterRuleTransaction.AddUpdaterAction
- TUpdaterRuleTransaction.PerformClientSideValidation
- TUpdaterRuleTransaction.PopulateViolationAndErrorInterfaceClasses
- TUpdaterRuleTransaction.Set_BDE_Alias

- TUpdaterRuleTransaction.Set_ClientLocale
- TUpdaterRuleTransaction.Set_ComputerID
- TUpdaterRuleTransaction.Set_MaxErrorCount
- TUpdaterRuleTransaction.Set_Password
- TUpdaterRuleTransaction.Set_POD
- TUpdaterRuleTransaction.Set_Priority
- TUpdaterRuleTransaction.Set_SequenceNum
- TUpdaterRuleTransaction.Set_SourceApplication
- TUpdaterRuleTransaction.Set_UserKey
- TUpdaterRuleTransaction.Set_Username

The following function was not disabled; however, it should not be called from a plug-in rule. Doing so would result in erroneous behavior.

- TUpdaterRuleTransaction.Submit

Typical steps for using Transaction to validate plug-in rules follow:

```
Parse ConfigString.   // Use later as necessary

for I := 0 to Transaction.UpdaterActionCount - 1 do

begin

Based on UpdaterAction[I].ActionType cast UpdaterAction[I] to one of
the following interface types:
       IuaGsegDelete
       IuaGsegInsert
       IuaGsegUpdate
       IuaDsegDelete
       IuaDsegInsert
       IuaDsegUpdate

Using the Iua… interface listed above, extract the segment
information.

Using the extracted segment information, configuration information,
and information gathered as necessary from the Aspect Workforce Web
services and DB, check for rule violations.

If rule violations occurred, call Transaction.AddStrictViolation(…)

If rule warnings are appropriate, call
Transaction.AddWarningViolation(…)

If an error occurred, call Transaction.AddError(…)

.

End
```

The interaction diagram for Updater plug-in rules is shown in Figure 3 3.

Interaction Diagram

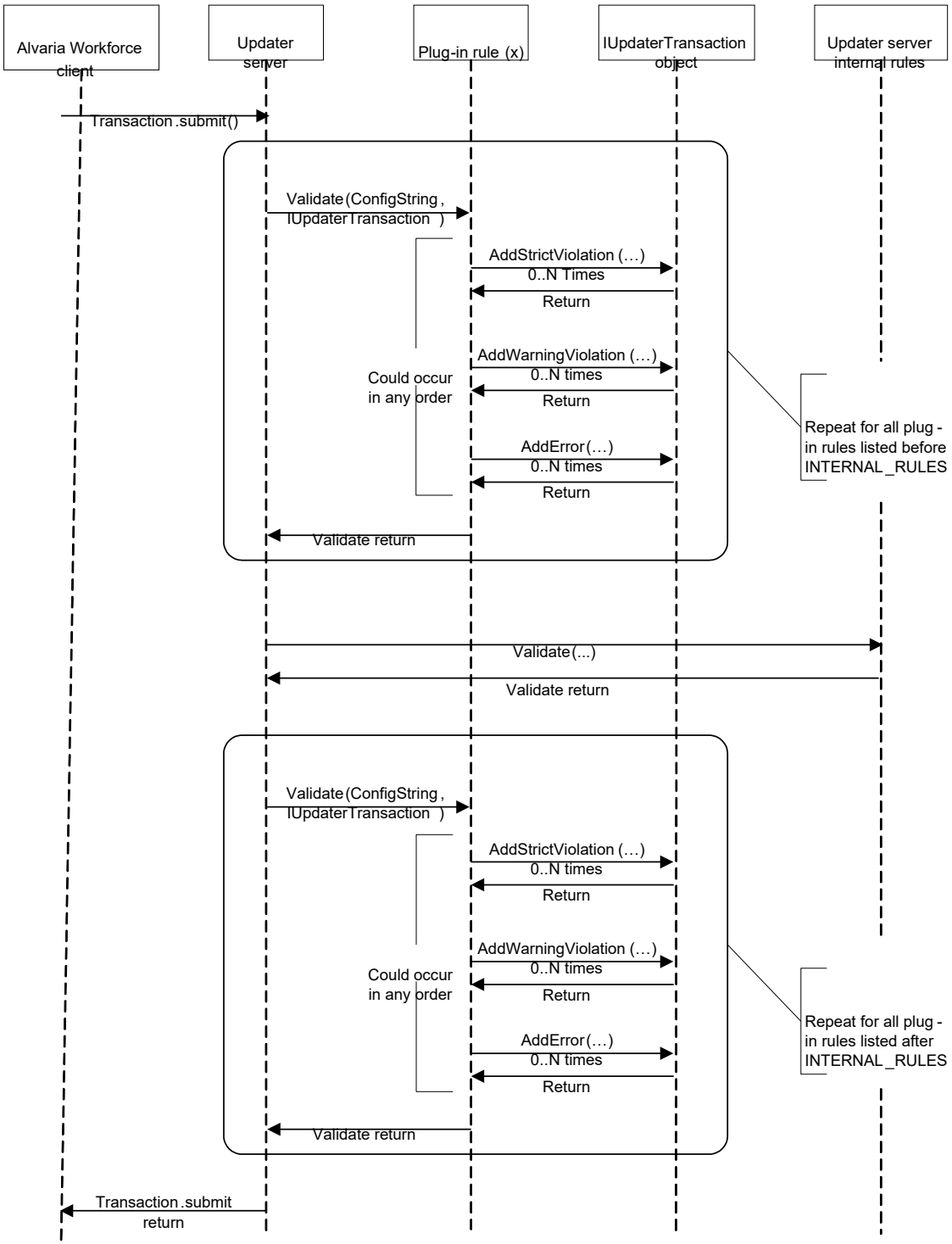


Figure 3: Updater Plug-In Rule Interaction DiagramWhen the “WFM Client” submits segment changes to the “Updater Server”, “Updater Server” would invoke both the “Plug-In Rule” and the “Updater Server Internal Rules” to determine if the segment changes contained in IUdaterTransaction were acceptable. It is important to understand the order of the invocation of the rule check. The order is governed by how the plug-in rules were setup via the Updater Configuration UI (see Invoking the Plug-In Rule). The plug-in

rules listed in the Updater Configuration UI before INTERNAL_RULES would be invoked prior to checking the internal rules. As one would expect, the plug-in rules listed after INTERNAL_RULES would be invoked after checking the internal rules. Also notice that when Validate(…) is invoked, the plug-in rule checks all segment changes listed in IUpdaterTransaction. This is why AddStrictViolation, AddWarningViolation, and AddError could be invoked multiple times. Each invocation would be for a separate segment change.

Error Messages

The IUpdaterTransaction functions AddStrictViolation(…), AddWarningViolation(…), AddError(…) take an ErrorMessage input argument. The ErrorMessage is logged via Updater's normal logging facilities. The developer of the plug-in rule is responsible for localizing ErrorMessage. The IUpdaterTransaction ClientLocale property contains XML that represents the locale information that should be used in representing the ErrorMessages. See the IUpdaterTransaction::ClientLocale property description for information on the content of the XML.

Type Library

The name and location of the IUpdaterTransaction type library follows:

- Name: UpdaterClient.tlb
- Typical Location: C:\Program Files\Common Files\Alvaria Shared

Additionally, section 0, describes how to use this type library via Visual Basic .Net.

Performance Guidelines

Performance is a key consideration when implementing plug-in rules. The maximum amount of time that should be allocated to any rule is dependant on the particular implementation. For example, if a site invokes make official and has a large number of agents over a large number of nominal days, then only a few seconds per employee per nominal date would be reasonable for the plug-in rule check. Conversely, smaller number of agents over fewer nominal days would allow more time for each plug-in rule. In general, the time to perform one check should be determined and that value should be used to determine the total time that would be consumed by the plug-in rule to make official at the particular site.

Installing and Configuring Plug-In Rules

This section addresses how to install and configure plug-in rules.

Installing the DLL

To implement Updater plug-in rules, first, install the plug-in rule DLL.

To install the plug-in rule DLL:

1. Copy the 64-bit and 32-bit DLLs to the following directories:
   C:\Program Files\Common Files\Alvaria Shared for the 64-bit version and C:\Program Files (x86)\Common Files\Alvaria Shared for the 32-bit version.



Note: The actual location may vary depending on the installation directory for Aspect Workforce.

2. Register the DLL according to your programming language. C++ COM DLLs are often registered using regsvr32.exe. .Net COM DLLs are registered using regasm.exe. The 64-bit version of regsvr32.exe is located in C:\Windows\System32. The 32-bit version of regsvr32.exe is located in C:\Windows\SysWOW64. The 64-bit version of regasm.exe is located in C:\Windows\Microsoft.Net\Framework64\v4.0.30319. The 32-bit version of regasm.exe is located in C:\Windows\Microsoft.Net\Framework\v4.0.30319.

Configuring Updater for Plug-In Rules

Configuring the plug-in rule entails identifying to Updater the Plug-In Rule DLL, and indicating when the rule should be invoked.

Identifying the Plug-In Rule DLL

A description of the steps necessary to identify the plug-in rule DLL to the Aspect Workforce system follows.

To identify the Updater plug-in rule DLL:

1. Open the Alvaria Service Installer by selecting Start > All Programs > Alvaria > Workforce > WFM Service Installer. The Alvaria Service Installer window opens.

2. In the main menu, select **Edit > Configure Updater Rules.** The **Configure Updater Rules** window opens.

3. Click **Add.** The **Updater Rule** dialog box opens.

4. In the Name field, type the **ProgId** of your new Updater rule.

5. In the Description field, type a **description** for the ProgId, and click **OK**. Your new Updater rule appears in the list of rules in the Configure Updater Rules window.

6. Click **OK** to close the Configure Updater Rules window, and close the **Alvaria Service Installer**.

Invoking the Plug-In Rule

The previous section demonstrated how to identify the Plug-In Rule DLL to the Aspect Workforce system. This section entails specifying that the plug-in rule should be invoked, as well as the order of the invocation relative to other rules.

Specifying Invocation of the Plug-In Rules

Use the Alvaria Service Installer to instruct the Aspect Workforce system that a plug-in role is to be invoked by the system.

To specify invocation of a plug-in rule:

1. Open the Alvaria Service Installer by selecting Start > All Programs > Alvaria > Workforce > WFM Service Installer. The Alvaria Service Installer window opens.

2. In the left pane, in the list of services, select **WFM Updater.**

3. In the main menu, select **Edit > Edit.** The **WFM Updater** window opens.

4. Click **Add.** The **Select Updater Rule** dialog opens.

5. Select a **rule** that you want the system to invoke, and click **OK.** The **WFM Updater** window opens.

6. Repeat step 4 and step 5 for each plug-in rule to be invoked.

Specifying the Invocation Order

Specify the invocation order to define the sequence in which Aspect Workforce executes Updater plug-in rules and internal rules. Internal rules are the native Aspect Workforce rules that Updater always executes, even if there are no Updater plug-in rules specified. Updater plug-in rules are evaluated in a specific order relative to each other and relative to the internal rules of Aspect Workforce.

To specify the invocation order of Updater plug-in rules:

1. Specify the **Updater plug-in rules** that you want to invoke. For instructions, see Specifying Invocation of the Plug-In Rules.

2. In the WFM Updater window, change the order of the rules by selecting a **rule** and dragging it to another position in the list.


Note: Rules are evaluated in order from top to bottom in the list.

3. Click **OK.**

4. Close the **Alvaria Service Installer.**

Configuring Plug-In Rules

The plug-in rule configuration information is that information that is specific to the operation of the plug-in rule. For example, if the plug-in rule limits the shift length to some maximum value, then that value should be a configuration parameter.

The plug-in DLL must have a GetConfigurationInfo function (see GetConfigurationInfo). This function normally displays a UI for the purposes of entering the configuration information. For the previously mentioned example, the UI would prompt for the maximum segment length. The configuration information would be persisted in the Admin DB, as described later in this section. When the Validate function (see Validate) is invoked, this string would be passed as one of the arguments. The plug-in rule DLL would parse the string to determine the value for the maximum shift length. This value would be used in the checks performed by the Validate function. The format of the configuration data is user defined. For the maximum shift length example, it could be a string presentation of an integer, XML formatted data, or any other format desired by the plug-in rule author.

To configure an Updater plug-in rule:

1. Open the Alvaria Service Installer by selecting Start > All Programs > Alvaria > Workforce > WFM Service Installer. The Alvaria Service Installer window opens.

2. In the left pane, in the list of services, select **WFM Updater.**

3. In the main menu, select **Edit > Edit.** The **WFM Updater** window opens.

4. Click **Configure.** A dialog opens with the name of the Updater plug-in rule in the title bar.

5. In the Timeout (Sec) field, enter the **timeout** value.

The timeout value is the time allotted for each employee/nominal date pair. For example, if the segment changes contain twenty employee/nominal date pairs, then the time allotted for the Validate function would be twenty times the value entered in the above screen. If the plug-in rule takes more time than

allotted, Updater would discontinue the plug-in rule check and would consider the segment changes to have failed the plug-in rule check.

Select the lookup button on the right side of the Configuration area. A dialog box opens with the name of the plug-in rule in the title bar.

Selecting the lookup button invokes the GetConfigurationInfo function for the plug-in rule. Typically the GetConfigurationInfo function presents a UI that enables the entry of configuration information. (For more information about the GetConfigurationInfo function, see GetConfigurationInfo.) When the GetConfigurationInfo function returns, the configuration data is shown in the Configuration area of the previous screen and is persisted in the Admin DB after clicking Save in the Alvaria Service Installer.



Note: The Alvaria Service Installer is an English-only application. It would be possible to localize the plug-in rule configuration UI; however, it might not be desirable to do so.

Sample Program

Following is a portion of a plug-in rule sample program written in Visual Basic 6. A more complete example written in Visual Basic .Net can be found in the software files included with this SDK. The name of the file is Updater Plug-In Rule Demo.zip. For more information, see Related Software Files.

```
Class ID = PluggableDemo.clsPluggable

Implements IWFMUpdaterRule

…

Private Sub IWFMUpdaterRule_GetConfigurationInfo(ByVal DBAlias As
String, ConfigString As String)
       ParseString ConfigString

strDBAlias = DBAlias

frmConfig.Show 1
       ConfigString = strWebServer & "|" & strWebUser & "|" &
       strWebPwd

End SubPrivate Sub IWFMUpdaterRule_Validate(ByVal ConfigString As
String, ByVal transaction As UpdaterClient.IUpdaterTransaction)
       Dim insertAction As UpdaterClient.IuaDsegInsert
       Dim insertActionG As UpdaterClient.IuaGsegInsert
       Dim dEmpSK As Double
       Dim dSegCodeSK As Double
       Dim lNomdate As Long
       Dim dtBalDate As Date
       Dim lBalance As Long
       On Error Resume Next
       ParseString ConfigString
       'Check to see if Detail else General
       If transaction.UpdaterActions(0).ActionType =
       UActionType_GsegInsert Then
              Set insertActionG = transaction.UpdaterActions(0)
              dEmpSK = insertActionG.New_Emp_SK
              lNomdate = insertActionG.New_Nom_Date
              dSegCodeSK = insertActionG.New_Seg_Code_SK
       Else
              Set insertAction = transaction.UpdaterActions(0)
              dEmpSK = insertAction.New_Emp_SK
              lNomdate = insertAction.New_Nom_Date
              dSegCodeSK = insertAction.New_Seg_Code_SK
       End If
       'Check if Segment Code is Vacation
```

```
        If GetSegCode(Trim(Str(dSegCodeSK))) = "VACA" Then
               dtBalDate = lNomdate
               lBalance = GetPABalance(Trim(Str(dEmpSK)), lNomdate)
               'Check if Vacation balance is greater than Zero
               If lBalance <= 0 Then
                   transaction.AddStrictViolation dSegCodeSK,
                   Bool_True, dEmpSK, lNomdate, 0, Bool_False, 0, 0,
                   65563, "Vacation Balance is less than 0: " & _
                   lBalance & vbCrLf & _
                   " for Employee SK " & Str(dEmpSK) & vbCrLf & _
                   " as of Date: " & dtBalDate & vbCrLf & _
                   " SegCode " & "VACA" & vbCrLf & _
                   " SegCode SK " & Str(dSegCodeSK) & vbCrLf & _
                   strTest & " " & vbCrLf & strTest1
               End If
        End If
        Set insertActionG = Nothing
        Set insertAction = Nothing

End Sub
```

Notification Server API Developer Guide

This guide documents Notification Server, an application in Empower. The APIs provided by Notification Server for use in developing custom applications are fully described in terms of properties and methods.

To use the full capabilities of the Notification Server APIs, an Aspect Workforce database must be installed.

Please note that starting with version 8.2 of Aspect Workforce all of the Notification Server related services now run as 64 bit applications. This means that COM components intended to use the Notification Server API must be built to run in a 64 bit environment.

Related Software Files

The SDK includes three example programs for this API, each of which shows the proper usage of one or more Notification Server interfaces. After you install the SDK, these files can be found at the following default path:

```
C:\Program Files (x86)\Alvaria\Workforce 25.0 SDK\API_Related_Files\
Notification_Server_API_Related_Files
```

Table 1 lists and briefly describes the example programs.

Table 1: Notification Server API Example Programs

Name	Interfaces Included
Example1	IAsptContactInfo, IAsptContactInfoProvider
Example2	IAsptNotification, IAsptNotificationSubmissionProvider
Example3	IAsptNotification, IAsptNotificationManagementProvider

Prerequisites

Table 2 shows the requirements for using the Notification Server API.

Table 2: Aspect Workforce Requirements for the Notification Server APIs

Versions	Aspect Workforce 7.5 or later
Enhancement Packages	Empower 7.5 or later
Database	Recommended

Audience

This guide is intended for developers who are extending the capabilities of Notification Server to support additional data sources and media types. This document assumes you are skilled with the following:

- A programming environment that supports COM/COM+
- Aspect Workforce

About Notification Server

Notification Server is an application that helps to manage staffing-related changes in the contact center. The primary value added by Notification Server is the automatic issuance of staffing-related change notifications to affected parties.

Notification Server provides several COM interfaces that allow the product to be extended to support additional data sources and media types. Each of these COM interfaces provides support for automation. Each interface is described in this guide with its associated properties and methods.

About the Notification Server API

This section lists the features and interfaces of the Notification Server API. It also provides tables showing the properties and methods for each interface, and provides information about the example programs included on the CD.

List of Features

The Notification Server API contains the following features:

- The capability to interface Notification Server to any employee contact information store via the IAsptContactInfoProvider and IAsptContactInfo COM interfaces.
- The capability to interface Notification Server to custom notification event generators via the IAsptNotificationSubmissionProvider, IAsptNotification, and IAsptNotificationAttachment COM interfaces.
- The capability to extend Notification Server to support additional database products via the IAsptNotificationSubmissionProvider and IAsptNotificationManagementProvider COM interfaces.
- The capability to extend Notification Server to support additional media types via the IAsptNotificationDispatcher and IAsptNotificationXMLGenerator COM interfaces.
- The capability to extend Notification Server to support additional mechanisms for communicating with dispatchers via the IAsptNotificationStream and IAsptNotificationStreamProvider COM interfaces.

List of Interfaces

The Notification Server API contains the following interfaces. A description, properties, and methods of each interface are documented in the sections that follow.

- IAsptContactInfoProvider
- IAsptContactInfo
- IAsptNotificationSubmissionProvider
- IAsptNotificationManagementProvider

- IAsptNotification
- IAsptNotificationAttachment
- IAsptNotificationDispatcher
- IAsptNotificationStream
- IAsptNotificationStreamProvider
- IAsptNotificationXMLGenerator

List of Properties and Methods by Interface

The following tables show the properties and methods for each interface.

IAsptContactInfoProvider Interface

IAsptContactInfoProvider Interface Properties and Methods

Property	Method
UserName	GetContactInfo
Password	Open
ConnectionName	Close
CLSID	
DBAlias	

IAsptContactInfo Interface

IAsptContactInfo Interface Properties and Methods

Property	Method
ContactID	
ContactProperty	

IAsptNotificationSubmissionProvider Interface

IAsptNotificationSubmissionProvider Interface Properties and Methods

Property	Method
NotificationStatus	CreateNotification
NumNotificationsHint	SubmitNotification
ConnectionName	Open
	Close

Property	Method
	CreateAttachment
	SubmitAttachment

IAsptNotificationManagementProvider Interface

IAsptNotificationManagementProvider Interface Properties and Methods

Property	Method
NumNewNotifications	Open
Notification	GetNewNotification
NotificationStatus	Close
UserName	UpdateNotificationProcTs
Password	UpdateNotificationDispatchID
ConnectionName	UpdateNotificationStatusAndRetryCount
NumAttachments	GetNextAttachment

IAsptNotification Interface

IAsptNotification Interface Properties and Methods

Property	Method
NotificationID	AddRecipientID
Status	RemoveRecipient
SubmissionDate	RemoveDuplicateRecipients
LastProcessedDate	
DispatcherID	
RetryCount	
RetryInterval	
ContactInfoProvider	
RecipientCount	
DispatchType	
Recipients	
NotificationMessage	

Property	Method
SenderID	

IAsptNotificationAttachment Interface

IAsptNotificationAttachment Interface Properties and Methods

Property	Method
NotificationAttachmentID	
NotificationID	
FilePathname	
MediaType	
DispType	

IAsptNotificationDispatcher Interface

IAsptNotificationDispatcher Interface Properties and Methods

Property	Method
DispatcherType	DispatchNotification
WorkQueueSize	DispatchNotificationWithAttachment

IAsptNotificationXMLGenerator Interface

IAsptNotificationXMLGenerator Interface Properties and Methods

Property	Method
	GenerateXML
	GenerateXMLWithAttachment

IAsptNotificationStreamProvider Interface

IAsptNotificationStreamProvider Interface Properties and Methods

Property	Method
CLSID	CreateStream

IAsptNotificationStream Interface

IAsptNotificationStream Interface Properties and Methods

Property	Method
ReadProperty	WriteNotification

Interface Definitions

The following sections describe each interface and provide syntax, description, and return values for each property and method in the interface.

IAsptContactInfoProvider

The IAsptContactInfoProvider interface abstracts the connection to an employee contact information store. Different implementations of this interface may provide access to different employee contact stores. The interface defines all IDs (keys) to be of string type. This allows users of the IAsptContactInfoProvider interface to treat all providers identically. Only the IAsptContactInfoProvider implementation knows the true type (e.g. double) of IDs in the store it represents. The conversion from string to the actual data type of the ID in the contact store is performed by the IAsptContactInfoProvider implementation.

The properties and methods of the IAsptContactInfoProvider interface are fully defined in the remainder of this section.

IAsptContactInfoProvider::UserName Property

Syntax

```
[propget] HRESULT UserName( [out, retval] BSTR *pVal );
[propput] HRESULT UserName( [in] const BSTR newVal ) const;
```

Description

This read/write property specifies the username used to open a connection to the contact information store. This property must be set prior to calling IAsptContactInfoProvider::Open.

Return Values

IAsptContactInfoProvider::UserName Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success
Write	S_FALSE	When newVal is the same as the current value of UserName

IAsptContactInfoProvider::Password Property

Syntax

```
[propput] HRESULT Password( [in] const BSTR newVal ) const;
```

Description

This write-only property specifies the password used to open a connection to the contact information store. This property must be set prior to calling IAsptContactInfoProvider::Open.

Return Values

IAsptContactInfoProvider::Password Property Return Values

Read/Write	Result	Condition
Write	S_OK	Success
Write	S_FALSE	When newVal is the same as the current value of Password

IAsptContactInfoProvider::ConnectionName Property

Syntax

```
[propget] HRESULT ConnectionName( [out, retval] BSTR *pVal );
[propput] HRESULT ConnectionName( [in] BSTR newVal );
```

Description

This read/write property specifies the connection name used to open a connection to the contact information store. In general, the connection name will be a Windows registry key and connection information for the contact information store will be loaded when this property is set. This property must be set prior to calling IAsptContactInfoProvider::Open.

Return Values

IAsptContactInfoProvider::ConnectionName Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success
Write	S_FALSE	When newVal is the same as the current value of ConnectionName
Write	E_INVALIDARG	When connection name is invalid (e.g. not found in the Windows registry)

IAsptContactInfoProvider::CLSID Property

Syntax

```
[propget] HRESULT CLSID( [out, retval] BSTR *pVal );
```

Description

This read-only property specifies the COM CLSID for the component object implementing the IAsptContactInfoProvider interface.

Return Values

IAsptContactInfoProvider::CLSID Property

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptContactInfoProvider::DBAlias Property

Syntax

```
[propget] HRESULT DBAlias( [out, retval] BSTR *pVal );
```

Description

This read-only property specifies the current Workforce Database Alias for the component object implementing the IAsptContactInfoProvider interface.

Return Values

IAsptContactInfoProvider::DBAlias Property

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptContactInfoProvider::GetContactInfo Method

Syntax

```
HRESULT GetContactInfo( [in] const BSTR ContactID,
[out, retval] IAsptContactInfo **ppContactInfo) const;
```

Description

This method returns an IAsptContactInfo interface for the contact specified by ContactID. An implementation of IAsptContactInfoProvider may cache instances of IAsptContactInfo, but this is not required. If an appropriate IAsptContactInfo instance is not available in cache, the IAsptContactInfoProvider implementation must create and fully populate an instance. The IAsptContactInfoProvider::Open method must be called prior to calling this method.

Return Values

IAsptContactInfoProvider::GetContactInfo Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The ppContactInfo parameter is NULL
_com_error	If a COM error occurs, the HRESULT value from the COM error is returned

IAsptContactInfoProvider::Open Method

Syntax

```
HRESULT Open();
```

Description

This method opens a connection to the contact information store. The UserName, Password, and ConnectionName properties must be set prior to calling this method.**Return Values**

Return Values

IAsptContactInfoProvider::Open Method

Result	Condition
S_OK	Success
S_FALSE	A connection is already open
E_INVALIDARG	The ConnectionName property was not successfully set prior to calling this method
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_FAIL	Unable to open a connection

IAsptContactInfoProvider::Close Method

Syntax

```
HRESULT Close();
```

Description

This method closes the connection to a contact information store. This method should be called prior to program termination for proper shutdown.

Return Values

IAsptContactInfoProvider::Close Method Return Values

Result	Condition
S_OK	Success
S_FALSE	A connection is not open
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting to close the connection

IAsptContactInfo

The IAsptContactInfo interface allows a client to access specific information about a particular contact. A client obtains an IAsptContactInfo instance from an IAsptContactInfoProvider instance via a call to the IAsptContactInfoProvider::GetContactInfo method. By providing an implementation of the IAsptContactInfo and IAsptContactInfoProvider interfaces, it is possible to interface Notification Server to any employee contact information store.

The properties of the IAsptContactInfo interface are fully defined in the remainder of this section.

IAsptContactInfo::ContactID Property

Syntax

```
[propget] HRESULT ContactID( [out, retval] BSTR *pVal ) const;
```

Description

This read-only property specifies the ID of the contact whose attributes may be read using this interface. The value of this property matches the value of the ContactID parameter in the call to IAsptContactInfoProvider::GetContactInfo that returned this interface.

Return Values

IAsptContactInfo::ContactID Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptContactInfo::ContactProperty Property

Syntax

```
[propget] HRESULT ContactProperty( [in] const BSTR PropertyName,
[out, retval] BSTR *pVal ) const;
```

Description

This read-only property allows a client to retrieve information about a particular contact. The PropertyName parameter specifies the name of the property to be retrieved. This method allows the list of contact information properties to be extending without changing the IAsptContactInfo interface. It is expected that the following properties will be supported by most implementations of IAsptContactInfo/IAsptContactInfoProvider:

- "LAST_NAME" for last name
- "FIRST_NAME" for first name
- "MIDDLE_NAME" for middle name
- "EMAIL" for email address
- "FAX" for fax number
- "PAGER" for pager number
- "PHONE" for phone number
- "SUPERVISOR-EMAIL-N" (N is 1 or more supervisor emails as configured in system)

Return Values

IAsptContactInfo::ContactProperty Property

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Read	E_INVALIDARG	The value of the PropertyName parameter is not supported

IAsptNotificationSubmissionProvider

The IAsptNotificationSubmissionProvider interface allows a client (e.g. a custom notification event generator) to submit notifications to Notification Server. The notifications will be persisted in the notification store, but this interface shields clients from having to know any details of the notification store whatsoever. In addition, this interface allows changes to be made to the notification store without impacts to clients or to the rest of Notification Server.

The properties and methods of the IAsptNotificationSubmissionProvider interface are fully defined in the remainder of this section.

IAsptNotificationSubmissionProvider::

NotificationStatus Property

Syntax

```
[propget] HRESULT NotificationStatus( [in] const double
NotificationID,

[out, retval] Aspt_Notification_Status *Status ) const;
```

Description

This read-only property allows a client to check the status of a notification. The NotificationID parameter specifies the notification whose status is returned in Status. The Aspt_Notification_Status type is defined below.

```
typedef
[
v1_enum,
version(1.0),
helpstring("Notification Status")
]
enum _Aspt_Notification_Status
{
Notification_Status_New = 0,
Notification_Status_Queued = 1,
Notification_Status_Pending_Dispatch = 2,
Notification_Status_Succeeded = 3,
Notification_Status_Failed = 4,
Notification_Status_Unknown = 5
}
```

Aspt_Notification_Status;

The IAsptNotificationSubmissionProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationSubmissionProvider::NotificationStatus Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The Status parameter is NULL
Read	_com_error	If a COM error occurs the HRESULT value from the COM error is returned
Read	E_FAIL	The status associated with NotificationID can not be retrieved for any reason

IAsptNotificationSubmissionProvider::

NumNotificationsHint Property

Syntax

```
[propput] HRESULT NumNotificationsHint( [in] unsigned long newVal );
```

Description

An implementation of IAsptNotificationSubmissionProvider must use some means to produce unique IDs to reference the individual notifications that are created over the life of the IAsptNotificationSubmissionProvider instance. There is some overhead associated with creating a unique ID. The NumNotificationsHint property provides a way for a client to specify the approximate number of IDs that will be needed. This allows the IAsptNotificationSubmissionProvider implementation to generate a number of IDs at once thus reducing the overhead.

Return Values

IAsptNotificationSubmissionProvider::NumNotificationsHint Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	_com_error	If a COM error occurs the HRESULT value from the COM error is returned
Read	E_FAIL	An error occurred while attempting to create unique IDs

IAsptNotificationSubmissionProvider::

ConnectionName Property

Syntax

```
[propget] HRESULT ConnectionName( [out, retval] BSTR *pVal );
[propput] HRESULT ConnectionName( [in] BSTR newVal );
```

Description

This read/write property specifies the connection name used to open a connection to the notification store. In general, the connection name will be a Windows registry key and connection information for the notification store will be loaded when this property is set. This property must be set prior to calling IAsptNotificationSubmissionProvider::Open.

Return Values

IAsptNotificationSubmissionProvider::ConnectionName Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success
Write	S_FALSE	When newVal is the same as the current value of ConnectionName
Write	E_INVALIDARG	When connection name is invalid (e.g. not found in the Windows registry)

IAsptNotificationSubmissionProvider::

CreateNotification Method

Syntax

```
HRESULT CreateNotification( [out, retval] IAsptNotification
**ppNotification );
```

Description

This method allows a client to create a new notification. IAsptNotification instances are typically not creatable on their own. Clients ask an IAsptNotificationSubmissionProvider instance to create an IAsptNotification instance for them, which they can then fill in with the appropriate values. IAsptNotification instances returned by this method have a valid NotificationID property and have the Status property initialized to Notification_Status_New. The IAsptNotificationSubmissionProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationSubmissionProvider::CreateNotification Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The ppNotification parameter is NULL
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting to create the notification

IAsptNotificationSubmissionProvider::

SubmitNotification Method

Syntax

```
HRESULT SubmitNotification( [in] const IAsptNotification
*pNotification );
```

Description

This method allows a client to submit a notification to be persisted to the notification store and dispatched. The client generally obtains the IAsptNotification instance via a call to IAsptNotificationSubmissionProvider::CreateNotification. The client should set the appropriate IAsptNotification properties prior to calling this method. The IAsptNotificationSubmissionProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationSubmissionProvider::SubmitNotification Method Return Values

Result	Condition
S_OK	Success
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting to persist the notification to the notification store

IAsptNotificationSubmissionProvider::

CreateAttachment Method

Syntax

```
HRESULT CreateAttachment( [out, retval] IAsptNotificationAttachment
**ppNotificationAttachment );
```

Description

This method allows a client to create a new notification attachment. IAsptNotificationAttachment instances are typically not creatable on their own. Clients ask an IAsptNotificationSubmissionProvider instance to create an IAsptNotificationAttachment instance for them, which they can then fill in with the appropriate values. IAsptNotificationAttachment instances returned by this method have a valid NotificationAttachmentID property. The IAsptNotificationSubmissionProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationSubmissionProvider::CreateAttachment Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The ppNotificationAttachment parameter is NULL
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting to create the attachment

IAsptNotificationSubmissionProvider::

SubmitAttachment Method

Syntax

```
HRESULT SubmitAttachment( [in] const IAsptNotificationAttachment
*pNotificationAttachment );
```

Description

This method allows a client to submit an attachment to be persisted to the notification store and dispatched with a notification. The client generally obtains the IAsptNotificationAttachment instance via a call to IAsptNotificationSubmissionProvider::CreateAttachment. The client should set the appropriate IAsptNotificationAttachment properties prior to calling this method. The IAsptNotificationSubmissionProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationSubmissionProvider::SubmitAttachment Method Return Values

Result	Condition
S_OK	Success
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting to persist the attachment to the notification store

IAsptNotificationSubmissionProvider::Open Method

Syntax

```
HRESULT Open();
```

Description

This method opens a connection to the notification store. The ConnectionName property must be set prior to calling this method.

Return Values

IAsptNotificationSubmissionProvider::Open Method Return Values

Result	Condition
S_OK	Success
S_FALSE	A connection is already open
E_INVALIDARG	The ConnectionName property was not successfully set prior to calling this method
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_FAIL	Unable to open a connection

IAsptNotificationSubmissionProvider::Close Method

Syntax

```
HRESULT Close();
```

Description

This method closes the connection to the notification store. This method should be called prior to program termination for proper shutdown.

Return Values

IAsptNotificationSubmissionProvider::Close Method Return Values

Result	Condition
S_OK	Success
S_FALSE	A connection is not open
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting to close the connection

IAsptNotificationManagementProvider

The IAsptNotificationManagementProvider interface provides the capability to manage the notification store. This interface shields clients from having to know any details of the notification store whatsoever. In addition, this interface allows changes to be made to the notification store without impacts to clients or to the rest of Notification Server

The properties and methods of the IAsptNotificationManagementProvider interface are fully defined in the remainder of this section.

IAsptNotificationManagementProvider::

NumNewNotifications Property

Syntax

```
[propget] HRESULT NumNewNotifications( [out, retval] long *pVal )
const;
```

Description

This read-only property specifies the number of new notifications (i.e. notifications with a Status of Notification_Status_New) in the notification store. The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::NumNewNotifications Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

Read/Write	Result	Condition
Read	_com_error	If a COM error occurs the HRESULT value from the COM error is returned
Read	E_UNEXPECTED	An error occurs when attempting to access the notification store

IAsptNotificationManagementProvider::Notification Property

Syntax

```
[propget] HRESULT Notification( [in] const double NotificationID,
[out, retval] IAsptNotification **ppNotification);
```

Description

This method returns an IAsptNotification instance for the notification specified by NotificationID. The IAsptNotificationManagementProvider implementation must create and fully populate the instance. The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::Notification Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The ppNotification parameter is NULL
Read	_com_error	If a COM error occurs the HRESULT value from the COM error is returned
Read	E_INVALIDARG	A notification corresponding to NotificationID is not found in the notification store
Read	E_UNEXPECTED	An error occurs when attempting to access the notification store

IAsptNotificationManagementProvider::NotificationStatus Property

Syntax

```
[propput] HRESULT NotificationStatus( [in] const double
NotificationID,
[in] Aspt_Notification_Status Status);
```

Description

This write-only property allows a client to set the status of the notification specified by NotificationID to the value specified by Status. The Aspt_Notification_Status type is defined below.

typedef

```
[
v1_enum,
version(1.0),
helpstring("Notification Status")
]
enum _Aspt_Notification_Status
{
Notification_Status_New = 0,
Notification_Status_Queued = 1,
Notification_Status_Pending_Dispatch = 2,
Notification_Status_Succeeded = 3,
Notification_Status_Failed = 4,
Notification_Status_Unknown = 5
} Aspt_Notification_Status;
```

The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::NotificationStatus Property Return Values

Read/Write	Result	Condition
Write	S_OK	Success
Write	_com_error	If a COM error occurs the HRESULT value from the COM error is returned
Write	E_UNEXPECTED	An error occurs when attempting to update the notification store

IAsptNotificationManagementProvider::UserName Property

Syntax

```
[propget] HRESULT UserName( [out, retval] BSTR *pVal );
[propput] HRESULT UserName( [in] const BSTR newVal ) const;
```

Description

This read/write property specifies the username used to open a connection to the notification store. This property must be set prior to calling IAsptNotificationManagementProvider::Open.

Return Values

IAsptNotificationManagementProvider::UserName Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success
Write	S_FALSE	When newVal is the same as the current value of UserName

IAsptNotificationManagementProvider::Password Property

Syntax

```
[propput] HRESULT Password( [in] BSTR newVal );
```

Description

This write-only property specifies the password used to open a connection to the notification store. This property must be set prior to calling IAsptNotificationManagementProvider::Open.

Return Values

IAsptNotificationManagementProvider::Password Property Return Values

Read/Write	Result	Condition
Write	S_OK	Success
Write	S_FALSE	When newVal is the same as the current value of Password

IAsptNotificationManagementProvider::ConnectionName Property

Syntax

```
[propget] HRESULT ConnectionName( [out, retval] BSTR *pVal );
[propput] HRESULT ConnectionName( [in] BSTR newVal );
```

Description

This read/write property specifies the connection name used to open a connection to the notification store. In general, the connection name will be a Windows registry key and connection information for the notification store will be loaded when this property is set. This property must be set prior to calling IAsptNotificationManagementProvider::Open.

Return Values

IAsptNotificationManagementProvider::ConnectionName Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success

Read/Write	Result	Condition
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success
Write	S_FALSE	When newVal is the same as the current value of ConnectionName
Write	E_INVALIDARG	When connection name is invalid (e.g. not found in the Windows registry)

IAsptNotificationManagementProvider:: NumAttachments Property

Syntax

```
[propget] HRESULT NumAttachments( [in] const double NotificationID,
[out, retval] long *pVal ) const;
```

Description

This read-only property specifies the number of attachments in the notification store that are associated with the specified NotificationID. The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::NumAttachments Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Read	_com_error	If a COM error occurs the HRESULT value from the COM error is returned
Read	E_UNEXPECTED	An error occurs when attempting to access the notification store

IAsptNotificationManagementProvider::GetNewNotification Method

Syntax

```
HRESULT GetNewNotification( [in] const unsigned long TimeOut,
[out, retval] IAsptNotification **ppNotification );
```

Description

This method returns an IAsptNotification instance for an entry in the notification store that has a NotificationStatus equal to Notification_Status_New. The TimeOut parameter specifies the length of time in msecs that the client wishes to be blocked in the event that no “new” notifications are currently in the notification store. A TimeOut value of 0xFFFFFFFF indicates that this method should block the client until a “new” notification can be returned. There is no requirement for this method to return “new” notifications in any particular order. The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::GetNewNotification Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The ppNotification parameter is NULL
S_FALSE	No “new” notifications in the notification store
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurred accessing the notification store

IAsptNotificationManagementProvider::GetNextAttachment Method

Syntax

```
HRESULT GetNextAttachment( [in] const double NotificationID,
[out, retval] IAsptNotificationAttachment **ppNotificationAttachment
);
```

Description

This method returns the next IAsptNotificationAttachment instance associated with the specified NotificationID. When the IAsptNotificationManagementProvider.NumAttachments property is read, this method is reset to the first attachment associated with the specified NotificationID. When there are no more attachments to return, the ppNotificationAttachment parameter is set to NULL. The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::GetNextAttachment Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The ppNotificationAttachment parameter is NULL
S_FALSE	No “new” notifications in the notification store
_com_error	If a COM error occurs the HRESULT value from the COM error is returned

Result	Condition
E_UNEXPECTED	An error occurred accessing the notification store

IAsptNotificationManagementProvider::Open Method

Syntax

```
HRESULT Open();
```

Description

This method opens a connection to the notification store. The ConnectionName property must be set prior to calling this method.

Return Values

IAsptNotificationManagementProvider::Open Method Return Values

Result	Condition
S_OK	Success
S_FALSE	A connection is already open
E_INVALIDARG	The ConnectionName property was not successfully set prior to calling this method
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_FAIL	Unable to open a connection

IAsptNotificationManagementProvider::Close Method

Syntax

```
HRESULT Close();
```

Description

This method closes the connection to the notification store. This method should be called prior to program termination for proper shutdown.

Return Values

IAsptNotificationManagementProvider::Close Method Return Values

Result	Condition
S_OK	Success
S_FALSE	A connection is not open

Result	Condition
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting to close the connection

IAsptNotificationManagementProvider:: UpdateNotificationProcTs Method

Syntax

```
HRESULT UpdateNotificationProcTs( [in] double NotificationID );
```

Description

This method sets the last processed date of the notification in the notification store specified by the NotificationID parameter. The last processed date is set to the current time in the notification store. The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::UpdateNotificationProcTs Method Return Values

Result	Condition
S_OK	Success
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurred when accessing the notification store

IAsptNotificationManagementProvider:: UpdateNotificationDispatchID Method

Syntax

```
HRESULT UpdateNotificationDispatchID( [in] double NotificationID,
[in] double DispatchID );
```

Description

This method allows a client to set the dispatch ID of the notification specified by NotificationID to the value specified by DispatchID. The dispatch ID specifies the dispatcher that is used to process the notification. The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::UpdateNotificationDispatchID Method Return Values

Result	Condition
S_OK	Success
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting access the notification store

IAsptNotificationManagementProvider:: UpdateNotificationStatusAndRetryCount Method

Syntax

```
HRESULT UpdateNotificationStatusAndRetryCount( [in] double
NotificationID,
[in] Aspt_Notification_Status status, [in] unsigned long RetryCount
);
```

Description

The method allows a client to update the status and retry count for the notification specified by NotificationID. As a notification is processed, the retry count is decremented each time an error occurs when attempting to dispatch the notification. When the retry count reaches zero, the notification status is set to Notification_Status_Failed. The Aspt_Notification_Status type is defined below.

```
typedef
[
v1_enum,
version(1.0),
helpstring("Notification Status")
]
enum _Aspt_Notification_Status
{
Notification_Status_New = 0,
Notification_Status_Queued = 1,
Notification_Status_Pending_Dispatch = 2,
Notification_Status_Succeeded = 3,
Notification_Status_Failed = 4,
Notification_Status_Unknown = 5
} Aspt_Notification_Status;
```

The IAsptNotificationManagementProvider::Open method must be called prior to calling this method.

Return Values

IAsptNotificationManagementProvider::UpdateNotificationStatusAndRetryCount Method

Result	Condition
S_OK	Success
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurs when attempting access the notification store

IAsptNotification

The IAsptNotification interface encapsulates the information needed to submit and dispatch a notification. Custom event generators may be developed using the IAsptNotification interface in combination with the IAsptNotificationSubmissionProvider interface.

The properties and methods of the IAsptNotification interface are fully defined in the remainder of this section.

IAsptNotification::NotificationID Property

Syntax

```
[propget] HRESULT NotificationID( [out, retval] double *pVal ) const;
```

Description

This read-only property uniquely identifies the notification in the notification store. The value of notification ID is set automatically by the IAsptNotificationSubmissionProvider::CreateNotification method and cannot be change afterwards.

Return Values

IAsptNotification::NotificationID Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotification::Status Property

Syntax

```
[propget] HRESULT Status( [out, retval] Aspt_Notification_Status
*pVal ) const;
```

Description

This read-only property specifies the current status of the notification. A client may update a notification's status via the IAsptNotificationManagementProvider::NotificationStatus property. The Aspt_Notification_Status type is defined below.

```
typedef
```

```
[
v1_enum,
version(1.0),
helpstring("Notification Status")
]
enum _Aspt_Notification_Status
{
Notification_Status_New = 0,
Notification_Status_Queued = 1,
Notification_Status_Pending_Dispatch = 2,
Notification_Status_Succeeded = 3,
Notification_Status_Failed = 4,
Notification_Status_Unknown = 5
} Aspt_Notification_Status;
```

Return Values

IAsptNotification::Status Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotification::SubmissionDate Property

Syntax

```
[propget] HRESULT SubmissionDate( [out, retval] DATE *pVal ) const;
```

Description

This read-only property specifies the date/time when the notification was first persisted to the notification store. A notification's submission date is set automatically by the IAsptNotificationSubmissionProvider::SubmitNotification method.

Return Values

IAsptNotification::SubmissionDate Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotification::LastProcessedDate Property

Syntax

```
[propget] HRESULT LastProcessedDate( [out, retval] DATE *pVal )
const;
```

Description

This read-only property specifies the date/time when the notification was last processed. A notification's last processed date may be updated via the IAsptNotificationManagementProvider::UpdateNotificationProcTs method.

Return Values

IAsptNotification::LastProcessedDate Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotification::DispatcherID Property

Syntax

```
[propget] HRESULT DispatcherID( [out, retval] double *pVal ) const;
```

Description

This read-only property specifies the ID of the dispatcher processing the notification for dispatch. A notification's dispatcher ID may be set via the IAsptNotificationManagementProvider::UpdateNotificationDispatchID method.

Return Values

IAsptNotification::DispatcherID Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotification::RetryCount Property

Syntax

```
[propget] HRESULT RetryCount( [out, retval] long *pVal ) const;
```

Description

This read-only property specifies the number of times the dispatcher should try to dispatch the message. The dispatcher decrements the retry count after each failed dispatch attempt. If the retry count reaches zero the dispatcher sets the notification status to Notification_Status_Failed. A notification's retry count may be updated via the IAsptNotificationManagementProvider::UpdateNotificationStatusAndRetryCount method.

Return Values

IAsptNotification::RetryCount Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotification::RetryInterval Property

Syntax

```
[propget] HRESULT RetryInterval( [out, retval] long *pVal ) const;
[propput] HRESULT RetryInterval( [in] long newVal );
```

Description

This property specifies the number of seconds the dispatcher should wait between attempts to dispatch the message.

Return Values

IAsptNotification::RetryInterval Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotification::ContactInfoProvider Property

Syntax

```
[propget] HRESULT ContactInfoProvider( [out, retval] BSTR *pVal )
const;
[propput] HRESULT ContactInfoProvider( [in] BSTR newVal );
```

Description

This read/write property specifies the COM CLSID of the provider to be used to look up contact information for the notification. In general, this property should be set by an event generator prior to persisting the notification to the notification store via the IAsptNotificationSubmissionProvider::SubmitNotification method.

Return Values

IAsptNotification::ContactInfoProvider Property

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success

IAsptNotification::RecipientCount Property

Syntax

```
[propget] HRESULT RecipientCount( [out, retval] long *pVal ) const;
```

Description

This read-only property specifies the number of recipients for the notification. This property is automatically adjusted by the IAsptNotification implementation as recipients are added or removed.

Return Values

IAsptNotification::RecipientCount Property

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotification::DispatchType Property

Syntax

```
[propget] HRESULT DispatchType( [out, retval] BSTR *pVal ) const;
[propput] HRESULT DispatchType( [in] BSTR newVal );
```

Description

This read/write property specifies the dispatch type of the notification (for example, “smtp”, “fax”). In general, this property should be set by an event generator prior to persisting the notification to the notification store via the IAsptNotificationSubmissionProvider::SubmitNotification method.

Return Values

IAsptNotification::DispatchType Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success

IAsptNotification::Recipients Property

Syntax

```
[propget] HRESULT Recipients( [in] const long Index, [out, retval]
BSTR *pVal ) const;
```

Description

This read-only array property specifies the ID (or key) of the recipient specified by the Index parameter. The ID is a string and must be of a format understood by the contact information provider associated with the notification.

Return Values

IAsptNotification::Recipients Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Read	E_INVALIDARG	The specified Index is out of range

IAsptNotification::NotificationMessage Property

Syntax

```
[propget] HRESULT NotificationMessage( [out, retval] BSTR *pVal )
const;

[propput] HRESULT NotificationMessage( [in] BSTR newVal );
```

Description

This read/write property specifies the message that is to be dispatched to a notification's recipients. In general, this property should be set by an event generator prior to persisting the notification to the notification store via the IAsptNotificationSubmissionProvider::SubmitNotification method.

Return Values

IAsptNotification::NotificationMessage Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success

IAsptNotification::SenderID Property

Syntax

```
[propget] HRESULT SenderID( [out, retval] BSTR *pVal ) const;
[propput] HRESULT SenderID( [in] BSTR newVal );
```

Description

This read/write property specifies the ID of the sender of the notification. In general, this property should be set by an event generator prior to persisting the notification to the notification store via the IAsptNotificationSubmissionProvider::SubmitNotification method.

Return Values

IAsptNotification::SenderID Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Write	S_OK	Success

IAsptNotification::AddRecipientID Method

Syntax

HRESULT AddRecipientID([in] BSTR RecipientID);

Description

This method adds the recipient ID (or key) identified by the RecipientID parameter to the list of recipients for the notification. The recipient ID must be in a format understood by the contact information provider associated with the notification. In general, an event handler should add all recipients to the notification via this method prior to persisting the notification to the notification store via the IAsptNotificationSubmissionProvider::SubmitNotification method.

Return Values

IAsptNotification::AddRecipientID Method Return Values

Result	Condition
S_OK	Success
E_UNEXPECTED	An error occurred while adding the recipient ID

IAsptNotification::RemoveRecipient Method

Syntax

```
HRESULT RemoveRecipient( [in] long Index );
```

Description

This method removes the recipient ID corresponding to the specified Index from the notification.

Return Values

IAsptNotification::RemoveRecipient Method Return Values

Result	Condition
S_OK	Success
E_INVALIDARG	Index is out of range

IAsptNotification::RemoveDuplicateRecipients Method

Syntax

```
HRESULT RemoveDuplicateRecipients();
```

Description

This method removes any duplicate recipient IDs from the notification.

Return Values

IAsptNotification::RemoveDuplicateRecipients Method Return Values

Result	Condition
S_OK	Success
E_UNEXPECTED	An error occurred while removing duplicate IDs

IAsptNotificationAttachment

The IAsptNotificationAttachment interface encapsulates the information associated with a notification attachment, such as an email attachment.

The properties and methods of the IAsptNotificationAttachment interface are fully defined in the remainder of this section.

IAsptNotificationAttachment::NotificationAttachmentID Property

Syntax

```
[propget] HRESULT NotificationAttachmentID( [out, retval] double
*pVal ) const;
```

Description

This read-only property uniquely identifies the notification attachment in the notification store. The value of notification attachment ID is set automatically by the IAsptNotificationSubmissionProvider::CreateAttachment method and cannot be change afterwards.

Return Values

IAsptNotificationAttachment::NotificationAttachmentID Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotificationAttachment::NotificationID Property

Syntax

```
[propget] HRESULT NotificationID( [out, retval] double *pVal ) const;
[propput] HRESULT NotificationID( [in] double newVal) const;
```

Description

This property uniquely identifies the notification in the notification store with which the attachment is associated.

Return Values

IAsptNotificationAttachment::NotificationID Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotificationAttachment::FilePathname Property

Syntax

```
[propget] HRESULT FilePathname( [out, retval] BSTR *pVal ) const;
[propput] HRESULT FilePathname( [in] BSTR newVal) const;
```

Description

This property specifies the pathname at which the attachment will be temporarily saved as a file during notification processing.

Return Values

IAsptNotificationAttachment::FilePathname Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotificationAttachment::MediaType Property

Syntax

```
[propget] HRESULT MediaType( [out, retval] BSTR *pVal ) const;
[propput] HRESULT MediaType( [in] BSTR newVal) const;
```

Description

This property specifies the MIME Content Type of the attachment.

Return Values

IAsptNotificationAttachment::MediaType Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotificationAttachment::DispType Property

Syntax

```
[propget] HRESULT DispType( [out, retval] BSTR *pVal ) const;
[propput] HRESULT DispType( [in] BSTR newVal) const;
```

Description

This property specifies the MIME Content Disposition Type of the attachment. This is normally either “inline” or “attachment”.

Return Values

IAsptNotificationAttachment::DispType Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotificationDispatcher

The IAsptNotificationDispatcher interface encapsulates the last stage in the delivery of a notification. The implementation of this interface is the only component that knows (or needs to know) how a notification is dispatched to a particular type of media (for example, email, fax). Notification Server can be extended to support additional media types by providing implementations of the IAsptNotificationDispatcher and IAsptNotificationXMLGenerator interfaces.

The properties and methods of the IAsptNotificationDispatcher interface are fully defined in the remainder of this section.

IAsptNotificationDispatcher::DispatcherType Property

Syntax

```
[propget] HRESULT DispatcherType( [out, retval] BSTR *pVal ) const;
```

Description

This read-only property specifies the media type supported by the dispatcher (e.g. email, fax).

Return Values

IAsptNotificationDispatcher::DispatcherType Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotificationDispatcher::WorkQueueSize Property

Syntax

```
[propget] HRESULT WorkQueueSize( [out, retval] long *pVal ) const;
```

Description

This read-only property specifies the number of notifications in the dispatcher's queue pending dispatch.

Return Values

IAsptNotificationDispatcher::WorkQueueSize Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotificationDispatcher::DispatchNotification Method

Syntax

```
HRESULT DispatchNotification( [in] const BSTR NotificationID,
[in] BSTR NotificationDispatchType,
[in] BSTR Sender, [in] BSTR Recipients,
[in] BSTR NotificationMessage,
[out, retval] Aspt_Notification_Status *pStatus) const;
```

Description

This method allows a client to submit a notification to the dispatcher. The NotificationID parameter specifies the unique ID (or key) of the notification. The NotificationDispatchType parameter specifies the media type (e.g. email, fax) of the notification. The Sender parameter specifies the originator of the notification. The Recipients parameter specifies the recipients of the notification. The NotificationMessage parameter specifies the notification message and is generally produced via the IAsptNotificationXMLGenerator interface. The status of the notification dispatch is returned via the pStatus parameter. The Aspt_Notification_Status type is defined below.

```
typedef
[
v1_enum,
version(1.0),
helpstring("Notification Status")
]
enum _Aspt_Notification_Status
{
Notification_Status_New = 0,
Notification_Status_Queued = 1,
Notification_Status_Pending_Dispatch = 2,
Notification_Status_Succeeded = 3,
Notification_Status_Failed = 4,
Notification_Status_Unknown = 5
} Aspt_Notification_Status;
```

Return Values

IAsptNotificationDispatcher::DispatchNotification Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The pStatus parameter is NULL
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurred during dispatch

IAsptNotificationDispatcher::DispatchNotificationWithAttachment Method

Syntax

```
HRESULT DispatchNotificationWithAttachment( [in] const BSTR
NotificationID,
[in] BSTR NotificationDispatchType,
[in] BSTR Sender, [in] BSTR Recipients,
```

```
[in] BSTR NotificationMessage,
[in] BSTR Attachments,
[out, retval] Aspt_Notification_Status *pStatus) const;
```

Description

This method allows a client to submit a notification that has associated attachments to the dispatcher. The NotificationID parameter specifies the unique ID (or key) of the notification. The NotificationDispatchType parameter specifies the media type (e.g. email, fax) of the notification. The Sender parameter specifies the originator of the notification. The Recipients parameter specifies the recipients of the notification. The NotificationMessage parameter specifies the notification message and is generally produced via the IAsptNotificationXMLGenerator interface. The Attachments parameter specifies the attachments associated with the notification. The status of the notification dispatch is returned via the pStatus parameter. The Aspt_Notification_Status type is defined below.

```
typedef
[
v1_enum,
version(1.0),
helpstring("Notification Status")
]
enum _Aspt_Notification_Status
{
Notification_Status_New = 0,
Notification_Status_Queued = 1,
Notification_Status_Pending_Dispatch = 2,
Notification_Status_Succeeded = 3,
Notification_Status_Failed = 4,
Notification_Status_Unknown = 5
} Aspt_Notification_Status;
```

Return Values

IAsptNotificationDispatcher::DispatchNotificationWithAttachment Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The pStatus parameter is NULL
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurred during dispatch

IAsptNotificationXMLGenerator

The IAsptNotificationXMLGenerator interface allows a client to generate an XML document to be sent on to an associated dispatcher without knowledge of the specifics of the XML format. Different dispatchers

may be associated with different implementations of this interface. Notification Server can be extended to support additional media types by providing implementations of the IAsptNotificationDispatcher and IAsptNotificationXMLGenerator interfaces.

The methods of the IAsptNotificationXMLGenerator interface are fully defined in the remainder of this section.

IAsptNotificationXMLGenerator::GenerateXML Method

Syntax

```
HRESULT GenerateXML( [in] const IAsptNotification *pNotification,
[in] const IAsptContactInfoProvider *pContactInfoProvider,
[out, retval] BSTR *pVal ) const;
```

Description

This method allows a client to format a notification as an XML document to be sent to an associated dispatcher for delivery. The pNotification parameter is a pointer to the IAsptNotification instance to be dispatched. The pContactInfoProvider is a pointer to the IAsptContactInfoProvider instance that can be used to get contact information for the contacts in pNotification. The resultant XML document is returned via pVal.

Return Values

IAsptNotificationXMLGenerator::GenerateXML Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The pVal parameter is NULL
E_FAIL	Invalid pNotification or pContactInfoProvider parameter

IAsptNotificationXMLGenerator::GenerateXMLWithAttachment Method

Syntax

```
HRESULT GenerateXMLWithAttachment( [in] const IAsptNotification
*pNotification,
[in] const IAsptNotificationManagementProvider
*pNotificationManagementProvider,
[in] const IAsptContactInfoProvider *pContactInfoProvider,
[out, retval] BSTR *pVal ) const;
```

Description

This method allows a client to format a notification that has associated attachments as an XML document to be sent to an associated dispatcher for delivery. The pNotification parameter is a pointer to the IAsptNotification instance to be dispatched. The pContactInfoProvider is a pointer to the

IAsptContactInfoProvider instance that can be used to get contact information for the contacts in pNotification. The resultant XML document is returned via pVal.

Return Values

IAsptNotificationXMLGenerator::GenerateXMLWithAttachment Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The pVal parameter is NULL
E_FAIL	Invalid pNotification or pContactInfoProvider parameter

IAsptNotificationStreamProvider

The IAsptNotificationStreamProvider interface (along with IAsptNotificationStream interface) allows a client to issue notifications to a dispatcher with no knowledge of the underlying communications mechanism. Notification Server can be extended to support additional communication mechanisms by adding new implementations of the IAsptNotificationStreamProvider and IAsptNotificationStream interfaces.

The properties and methods of the IAsptNotificationStreamProvider interface are fully defined in the remainder of this section.

IAsptNotificationStreamProvider::CLSID Property

Syntax

```
[propget] HRESULT CLSID( [out, retval] BSTR *pVal )  const;
```

Description

This read-only property specifies the COM CLSID for the component object implementing the IAsptNotificationStreamProvider interface.

Return Values

IAsptNotificationStreamProvider::CLSID Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL

IAsptNotificationStreamProvider::CreateStream Method

Syntax

```
HRESULT CreateStream( [in] const BSTR DispatchType,
[in] const BSTR RemoteAddr,
[in] const BSTR ExtraConfigParams,
```

```
[out, retval] IAsptNotificationStream **ppNotificationStream);
```

Description

This method allows a client to create an IAsptNotificationStream instance. The DispatchType parameter indicates the type of dispatcher to which the IAsptNotificationStream instance is to be connected. The RemoteAddr parameter identifies the location of the dispatcher on the network and can be a string in any format that is supported by the specific implementation of IAsptNotificationStreamProvider. The ExtraConfigParams parameter is a string that can be used to pass in additional information specific to a particular implementation of IAsptNotificationStreamProvider. For example, a socket-based implementation may require that the port number be specified via the ExtraConfigParams parameter.

Return Values

IAsptNotificationStreamProvider::CreateStream Method Return Values

Result	Condition
S_OK	Success
E_POINTER	The ppNotificationStream parameter is NULL
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurred during stream creation

IAsptNotificationStream

The IAsptNotificationStream interface along with the IAsptNotificationStreamProvider interface) allows a client to issue notifications to a dispatcher with no knowledge of the underlying communications mechanism. A client receives an IAsptNotificationStream instance by calling the IAsptNotificationStreamProvider::CreateStream method. Notification Server can be extended to support additional communication mechanisms by adding new implementations of the IAsptNotificationStreamProvider and IAsptNotificationStream interfaces.

The properties and methods of the IAsptNotificationStream interface are fully defined in the remainder of this section.

IAsptNotificationStream::ReadProperty Property

Syntax

```
[propget] HRESULT ReadProperty( [in] const BSTR PropertyName, [out,
retval] BSTR *pVal ) const;
```

Description

This read-only property allows a client to read the properties of a dispatcher. The PropertyName parameter specifies the name of the property to be read. The value of the property is returned via the pVal parameter.

Return Values

IAsptNotificationStream::ReadProperty Property Return Values

Read/Write	Result	Condition
Read	S_OK	Success
Read	E_POINTER	The pVal parameter is NULL
Read	_com_error	If a COM error occurs the HRESULT value from the COM error is returned
Read	E_UNEXPECTED	An error occurs when attempting to read the property

IAsptNotificationStream::WriteNotification Method

Syntax

```
HRESULT WriteNotification( [in] const BSTR NotificationDoc,
[out, retval] Aspt_Notification_Status *pStatus);
```

Description

This method writes the notification specified by the NotificationDoc parameter to the dispatcher connected to the IAsptNotificationStream instance. The status of the Notification is returned via the pStatus parameter. The Aspt_Notification_Status type is defined below.

```
typedef
[
v1_enum,
version(1.0),
helpstring("Notification Status")
]
enum _Aspt_Notification_Status
{
Notification_Status_New = 0,
Notification_Status_Queued = 1,
Notification_Status_Pending_Dispatch = 2,
Notification_Status_Succeeded = 3,
Notification_Status_Failed = 4,
Notification_Status_Unknown = 5
} Aspt_Notification_Status;
```

Return Values

IAsptNotificationStream::WriteNotification Method Return Values

Result	Condition
S_OK	Success

Result	Condition
E_POINTER	The pStatus parameter is NULL
_com_error	If a COM error occurs the HRESULT value from the COM error is returned
E_UNEXPECTED	An error occurred while attempting to write to the stream

Work Rule API Developer Guide

This guide specifies the Work Rule API for development of third-party applications with Aspect Workforce. The Work Rule API enables you to implement software components intended to prevent assignment of schedules to agents that would violate unique scheduling rules.Please note that starting with version 8.2, most of Aspect Workforce now runs as a 64-bit application. This means that COM components intended to use the Work Rule API must be built to run in a 64-bit environment.

Related Software Files

The SDK includes related files that can demonstrate the functionality of the Work Rule API. After you install the SDK, these files can be found at the following default path:

```
C:\Program Files (x86)\Alvaria\Workforce 25.0
SDK\API_Related_Files\Work_Rule_API_Related_Files
```

File List

Related files delivered with Aspect Workforce:

```
WorkRuleInterface.tlb\Program Files\Common Files\Alvaria Shared
```

Related software files delivered with the SDK consist of the following:

- SampleDayScheduleProperties.cs
- SampleDayScheduleSetProperties.cs
- SampleEmployeeContext.cs
- SampleEvaluationContext.cs
- SampleScheduleContext.cs
- SampleScoreCalculator.cs
- SampleWorkRule.cs
- SampleWorkRuleConfig.cs
- SampleWorkRuleConfigEditor.cs
- SampleWorkRuleConfigEditor.Designer.cs
- SampleWorkRuleConfigEditor.resx
- SampleWorkRuleCredentials.cs
- Serialize.cs

Prerequisites

The following table shows the requirements for using the Work Rule API.

Versions	Aspect Workforce 7.5 or later
Enhancement Packages	None
Database	Aspect Workforce database must be installed
Other Software	• Aspect Workforce client must be installed

Additional Requirements

In addition to the prerequisites, the Work Rule API has the following requirements relating to hardware, software, connectivity, and security.

- A computer that meets the requirements for the Aspect Workforce client software.
- The Aspect Workforce client installed on the computer from where you will use the Work Rule API.
- Work Rule plug-in components are hosted by the Aspect Workforce client and are used by the Trial Schedule Manager for Automatic Assignment and by the Schedule Runs modules for Preference-Based Scheduling.
- Network connections to the Aspect Workforce database server and application server(s).
- An Aspect Workforce username and password for use with the Work Rule API. If you do not have an Aspect Workforce account, you must ask your administrator to create one for you.
- ODBC drivers for your Aspect Workforce database.

Audience

The Work Rule API can be used by third-party developers and others to create custom work rule plug-ins that are called during the automatic schedule assignment and preference-based scheduling processes. These work rule plug-ins can reject candidate schedule assignments and also provide some guidance to the scheduling algorithm to assist in prioritizing candidate schedules to attempt to avoid violating custom work rules.

This document assumes that you are familiar with the following topics:

- Basic database concepts
- SQL programming
- COM concepts and programming methods
- Microsoft .Net and C# (if you plan to follow the C# examples)
- Basic probability theory

For more information on these topics, please consult the documentation provided with the respective products or the large number of books available on these topics.

Additional Documentation

The documentation for the Work Rule API consists of this developer guide. However, general Aspect Workforce concepts are explained in the *Aspect Workforce User Guide*, and in the Aspect Workforce online Help.

Overview

The Work Rule API is designed to provide a mechanism to hook custom work rule logic into automatic schedule assignment and preference-based scheduling algorithms. Through the Work Rule API, a developer can implement plug-in components which can detect and reject schedule assignments that would violate work rules unique to their organization's business requirements, contractual labor requirements, or local labor laws.

In addition to rejecting schedule assignments, the work rule plug-in can provide guidance in prioritizing schedules based on the probability that the assignment will lead to a work rule violation in some future schedule assignment. For example, this guidance can help avoid situations where schedule assignments early in a month prevent many or all assignments late in the month.

Before using the Work Rule API, you must be familiar with the basic concepts of schedules and segments in Aspect Workforce. This document outlines some of the concepts that are fundamental in understanding schedules and the way that they are stored and manipulated in Aspect Workforce. You should also be familiar with the Automatic Assignment feature of the Trial Schedule Manager module and Add Preference- Based Schedules feature of the Schedule Runs modules within the Aspect Workforce client.

Components of the Work Rule API

The Work Rule API consists of the C# plug-in examples and this document.

Installing the Work Rule API

Installing the Work Rule API is a straightforward process. Perform the following steps on the computer where you want to access the Work Rule API.

1. Install the Aspect Workforce client software. This is done by running the Setup program from the WFMSetup share on your application server.

2. Copy the files included in this distribution into the directory containing the Aspect Workforce client files, typically C:\Program Files\Alvaria\Workforce.

3. Install the ODBC driver for your Aspect Workforce database.

4. Create an ODBC User or System DSN to connect to your Aspect Workforce database.

Concepts in Aspect Workforce

There are several fundamental Aspect Workforce concepts that you must understand before you can successfully use the Work Rule API. Some of these concepts are also explained in the Aspect Workforce online Help.

Please begin by understanding these earlier sections as the Work Rule API involves most of the same Aspect Workforce concepts:

- 0 Tally Server Concepts in Aspect Workforce
- 0 Updater API Concepts in Aspect Workforce

Trial Schedules

Trial schedules are schedules that have been generated by Aspect Workforce to meet some staffing requirements. Trial schedules may be in an assigned or unassigned state. Assigned trial schedules have

an associated employee while unassigned trial schedules have no relationship to a specific employee. The Automatic Assignment feature of the Trial Schedule Manager automatically assigns employees to unassigned trial schedules.

A single trial schedule may consist of segments associated with one or more nominal dates. For example, a trial schedule may consist of segments spanning an entire work week and is treated as a single multi-day schedule.

Assigned trial schedules are not yet an employee's official schedule. The Make Official command in the Trial Schedule Manager moves trial schedules to an employee's official schedule.

Official Schedules

Official schedules are made up of official segments which describe how an employee will actually spend their time.

Work Rules

A work rule is a constraint that employee schedules must satisfy to prevent assignments that would create undesirable or unfair work schedules over a period of days. Work rules are independent of employee schedule preferences. The constraint may arise from a variety of sources, such as labor agreements, legal or industry mandates, or business policies. Aspect Workforce includes software hooks that allow software components to enforce work rules (or approximations of work rules) during automatic assignment of trial schedules and preference-based scheduling.

Work Rule Sets

Work Rules Sets are collections of work rules that form a unit. The user of Automatic Assignment of Trial Schedules and Preference-Based Scheduling will select a single Work Rule Set to use to evaluate and constrain schedule assignments.

Work Rule Plug-ins

A work rule plug-in is a software component that implements the logic to evaluate and enforce a particular work rule or provides a general implementation for a family of work rules that vary by some set of parameters. Work rule plug-ins must implement the IWorkRule COM interface and adhere to the software contracts associated with that interface.

Aspect Workforce includes several work rule plug-ins that are useful to enforce a variety of work rules, and can be used to enforce approximations of others. Work rules that are included in Aspect Workforce are:

- Maximum Consecutive Schedules
- Minimum Consecutive Schedules
- Maximum Schedule(s) in Window
- Minimum Schedule(s) in Window
- Minimum Time Off After Schedule
- Minimum Continuous Time Off in Window
- Maximum Time in Window
- Minimum Time in Window

- Conditional Schedule(s)

Please consult *Aspect Workforce User Guide* and Aspect Workforce online help for more information regarding these standard work rule plug-ins. With some creative application of these work rule plug-ins, many common work rules may be enforced.

If a work rule cannot be sufficiently enforced by these work rule plug-ins, the Work Rule API may be used by software developers to implement plug-ins that enforce additional work rules and work rule families.

Work Rule Evaluation of Schedules

Work rules are given a set of one or more single-day schedules and must evaluate the schedules against the work rule. There are four outcomes of this evaluation:

1. Assigning this schedule will not violate a work rule and it will prevent the work rule from being violated with additional assignments.

2. Assigning this schedule will not violate a work rule but additional assignments may violate the work rule.

3. Assigning this schedule will violate a work rule but additional assignments may allow the rule to not be violated. (Work rule not satisfied)

4. Assigning this schedule will violate a work rule and additional assignment will also violate the work rule. (Work rule violation)

The work rule returns these results encoded in two parts.

5. A Boolean flag named Pass

    a. TRUE: The work rule is currently satisfied and the proposed schedule assignment should be allowed.

    b. FALSE: The work rule is not currently satisfied, but the proposed schedule could still be assigned depending on the value of the Score component

6. A score for the proposed schedule assignment which is the probability the rule will be satisfied given the schedule and all possible future schedule assignments

    a. Score is a probability which is a number falling between zero and one inclusive

    b. If Score is zero, there is no chance that the rule can pass and the proposed schedule will not be assigned

    c. If Pass is True and Score is between zero and one, then the rule is currently satisfied, but future schedule assignments could fail the rule.

    d. If Pass is False and Score is between zero and one, then the rule is not currently satisfied, but there is a chance future schedule assignments could satisfy the rule. This state often appears when we have a rule that specifies some minimum number of schedules or hours in some period. At first there may be no schedules, but with every assignment the rule is closer to being satisfied.

    e. If Score is one, then the rule is satisfied and no future schedule assignments will cause the rule to fail.

Work Rule Internal State

Work rules are evaluated in the innermost loops of the scheduling algorithm. For performance reasons, it is important the work rules take every opportunity to precompute and cache as much information as possible. Information that the work rule should store or cache include:

1. Work rule configuration

This will be provided to the work rule during initialization

2. A list of IDayScheduleSet properties

At the beginning of the schedule scope, the work rule will be notified of the list of candidate schedule sets. Each IDayScheduleSet in the list will contain one or more candidate IDaySchedule objects. Each IDayScheduleSet will be assigned to an agent as a unit. Multiple IDayScheduleSets may contain single-day schedules for the same nominal date, but only one will be assigned at a time.
The work rule should examine this list and precompute as much information as possible such as the pattern of on / off days, etc. This is because a IDayScheduleSet instance may be evaluated multiple times in different combinations, and if computations are performed for every combination, the algorithm performance will suffer greatly from combinatorial explosion.

3. Properties of the current employee

Schedules are assigned to one employee at a time. At the beginning of the employee scope, any required information for the current employee such as time zone should be loaded and cached.

4. Employee official schedules

Most work rules require the context of official schedules to evaluate any trial schedules. During initialization, the work rule is presented a date range for scheduling. From this date range, the work rule may need official schedules for some set of dates before and after the scheduling date range. When the employee scope begins, the work rule should take that opportunity to load all relevant official schedules for the given employee.

For example, if we have a work rule that limited overtime segments over a 7-day sliding window, and we had a scheduling date range from November 1 through November 14, the work rule would need to load all official schedules from October 26 through November 20 because those dates are required to provide the full 7 day window ending with November 1, and the 7 day window beginning with November 14, and all sliding 7- day windows in between. Unassigned official schedules (days off) must be assumed to be allowed to change to a working day in future schedule assignments. Because of this uncertainty we apply probability to model the likelihood that a work rule will pass or fail in the future.

5. The current set of “assigned” schedules (evaluation context)

Over the course of the scheduling run, the scheduling algorithm will select the next-best candidate schedule and ask the work rule to evaluate it. If the result is favorable, the schedule may be temporarily “assigned” by calling AddDayScheduleSet to add the schedule set to the work rule’s evaluation context. The work rule must hang on to this set of schedules and treat them as if they are official (replacing any official schedules with conflicting nominal dates) when evaluating subsequent schedules.

Any days without either an official schedule or an “assigned” schedule should be considered undefined. These nominal dates have a 50/50 probability of being assigned a schedule of interest in the future and any probability calculations should take undefined days into account. When the scheduling algorithm decides that a day will be a day off for the employee, it will “assign” a schedule without any general or detail segments. This is normal and the work rule plug-in should treat that nominal date as a day off with 100% probability.

The scheduling algorithm is a search with backtracking algorithm so the work rule may need to “unassign” candidate schedules when RemoveDayScheduleSet is called. If there are official schedules on a

removed nominal date, the official schedule becomes active again. Whenever the evaluation context changes through calls to AddDayScheduleSet or RemoveDayScheduleSet, the work rule should precompute as much as possible about the evaluation context to improve performance.

Work Rule Life Cycle during Scheduling

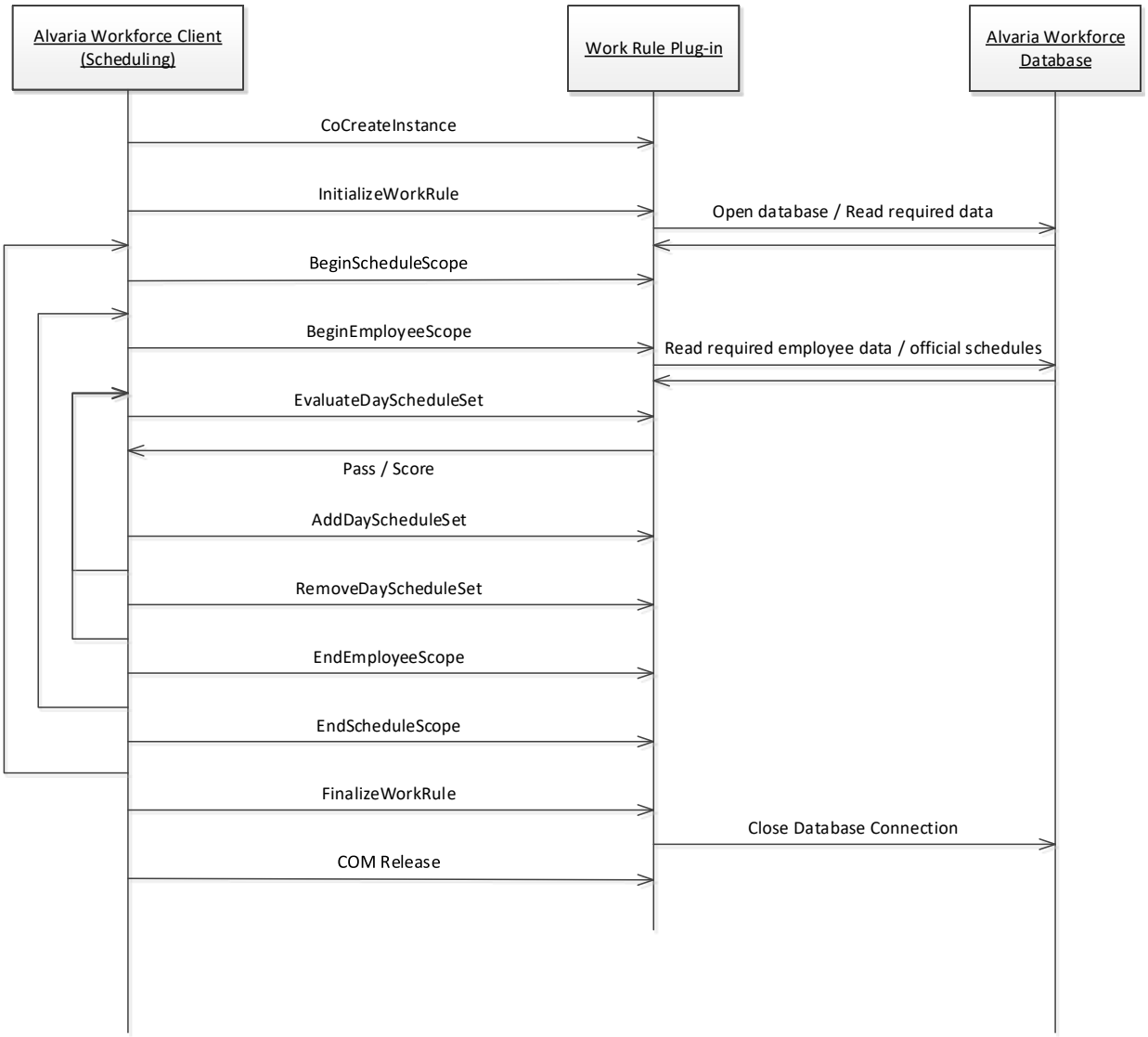


Figure 4: Work Rule Life Cycle Sequence Diagram

1. The scheduling algorithm creates an instance of the work rule plug-in COM object.
2. The scheduling algorithm calls InitializeWorkRule
    a. Validate configuration
    b. Cache configuration and encrypted credentials
    c. Connect to the database if necessary
    d. Connect to Aspect Workforce web services if necessary

3. The scheduling algorithm calls BeginScheduleScope

    a. Calculate and cache properties for each schedule in the schedule scope

    b. The work rule can assume that schedules prior to the scheduling window (official or provided via AddDayScheduleSet) will not change. That is to say, assigned schedules are fixed and unassigned schedules remain unassigned.

    c. The work rule must assume that any unassigned days within the scheduling window and in the future may be assigned a schedule and might not remain unassigned or time off. Alvaria work rules generally assume 50 / 50 probability that the date will be assigned a relevant schedule.

4. The scheduling algorithm calls BeginEmployeeScope

    a. Load required employee information

    b. Load and cache some window of official schedules as required

    c. Calculate and cache properties for each single-day official schedule

5. The scheduling algorithm calls EvaluateDayScheduleSet

    a. Given the set of official schedules, any assigned candidate schedules, and the rule configuration, calculate Pass and Score

    b. There are no state side effects in the work rule from EvaluateDayScheduleSet. The schedule is being evaluated, but not yet assigned

6. The scheduling algorithm calls AddDayScheduleSet

    a. When the algorithm decides a schedule is a good candidate for assignment, it calls AddDayScheduleSet

    b. The work rule should add the IDayScheduleSet to its set of assigned candidate schedules, replacing any overlapping official schedules

    c. The work rule should precompute whatever it can to represent the current set of assigned candidate schedules

7. The scheduling algorithm calls RemoveDayScheduleSet

    a. The scheduling algorithms are backtracking algorithms and will remove an assigned candidate schedule in order to test a different combination of IDayScheduleSet assignments.

    b. The work rule should remove the IDayScheduleSet from its set of assigned candidate schedules, reactivating any overlapping official schedules.

    c. The work rule should precompute whatever it can to represent the current state of assigned candidate schedules.

8. The scheduling algorithm calls EndEmployeeScope

    a. The scheduling algorithm is finished assigning schedules to an employee

    b. The work rule should clear its set of assigned schedules

c. The work rule should clear its set of official schedules

d. The work rule should clear any information specific to the employee

e. The work rule should expect to begin another employee scope

9. The scheduling algorithm calls EndScheduleScope

a. The scheduling algorithm is finished working with the current set of candidate schedules

b. The work rule should clear the cache of IDayScheduleSet properties

c. The work rule should expect to begin another schedule scope

10. The scheduling algorithm calls FinalizeWorkRule

a. The work rule should close any database and web service connections

b. The work rule should free any allocated resources

11. The work rule plug-in COM object is released

Automatic Assignment of Trial Schedules

The automatic assignment process tries to assign selected employees to selected trial schedules while satisfying each employee's schedule preferences and other parameters that you set. Automatic assignment focuses on one employee at a time, scans the trial schedules for a match, and goes to the next employee only when a match either is found or is not possible.

When work rules are enabled, automatic assignment allows the work rule plug-ins to inspect the trial schedule and either allow or disallow its assignment. If any work rule disallows the trial schedule, it will not be assigned to the employee. Work rule plug-ins should consider existing official schedules, as well as other assigned trial schedules, when determining if the next trial schedule is allowed or not.

Preference-Based Scheduling

Preference-Based Scheduling is an approach to generating trial schedules where employee preferences are considered in addition to staffing requirements. Trial schedules generated through Preference-Based Scheduling are automatically assigned to employees at creation time. Preference-Based Scheduling focuses on one employee at a time, generating a set of candidate trial schedules that are allowed by agent preferences. Preference-Based Scheduling then asks each work rule to score each trial schedule. If the score is zero, the trial schedule is discarded as being disallowed from assignment. These work rule scores are combined to help sort the schedules to favor schedules that are least likely to result in a rule violation in the future, while still meeting staffing objectives.

When work rules are enabled, preference-based scheduling allows the work rule plug-ins to inspect the trial schedule and either disallow its assignment or allow the assignment and provide a score. If any work rule disallows the trial schedule, it will not be assigned to the employee. Work rule plug-ins should consider existing official schedules, as well as other assigned trial schedules, when evaluating a new trial schedule.

Fiscal Calendars

Work rules are often defined to control occurrences of schedules within some range of dates. The date range may be defined in terms of either a Fiscal Calendar or a sliding date window. A Fiscal Calendar defines a set of named nominal date ranges. For example we may have a fiscal calendar named Years with the following ranges:

Fiscal Period	Start Nominal Date	Stop Nominal Date
2013	1/1/2013	12/31/2013
2014	1/1/2014	12/31/2014
2015	1/1/2015	12/31/2015

If not maintained, it is possible to reach the end of the list of fiscal periods. If this type of error occurs, the work rule plug-in should raise an exception or return an error HRESULT, depending on the language used to implement the plug-in. The user will be presented an error message and the scheduling run will stop.

Data Types and Stored Procedures

Please see section 0 Updater Data Types and Stored Procedures, as Work Rule Plug-ins depend on the same Aspect Workforce schedule data as Updater.

Database Tables

Please see section 0 Updater Database Tables, as Updater depends on many of the same database tables used to evaluate work rules.

RULE_PLUG_IN

The RULE_PLUG_IN table is where third-party work rule plug-ins are registered with Aspect Workforce. Each work rule plug-in requires an entry in this table, as well as being an installed and registered COM component on the Aspect Workforce client machine that will run the work rule plug-in.

Table 1: RULE_PLUG_IN Table

Column Name	Data Type	Description
RULE_PLUG_IN_SK	SK	Rule plug-in surrogate key
NAME	NVARCHAR(30)	NAME is the unique identifier of the rule plug-in. This will appear to the user configuring work rules in the rule plug-in list.
DESCR	NVARCHAR(30)	This is a description of the rule plug-in. This will appear to the user configuring work rules in the rule plug-in list.
SECUR_TYPE	NCHAR	This is the type of security model for the rule plug-in.

Column Name	Data Type	Description
		The only supported value for custom work rule plug-ins is ‘P’.
PROG_ID	NVARCHAR(256)	The COM ProgId of the coclass implementing the custom work rule plug-in. The COM DLL must be present and registered on all machines where the Aspect Workforce client will be used for scheduling.
SECUR_BLOB	NVARCHAR(max)	The work rule plug-in is responsible for collecting required credentials and returns them encrypted to the Aspect Workforce client. These encrypted credentials are stored in this column and returned to the work rule which will decrypt and use the credentials to access Aspect Workforce web services or the Aspect Workforce database. The work rule plug-in executes in-process with the Aspect Workforce client and may opt to use integrated security to access these resources.

FSC_CAL

The FSC_CAL table contains the primary record for each defined Fiscal Calendar. The period definitions are stored in FSC_CAL_DET. Work rules are often defined in terms of schedules within some range of dates. It is often convenient to define the range of dates in terms of fiscal calendars.

Column Name	DataType	Description
FSC_CAL_SK	SK	Surrogate key for the fiscal calendar
CODE	NVARCHAR(30)	The unique code identifier for the calendar
DESCR	NVARCHAR(30)	The description of the fiscal calendar
UPD_TCS_LOGIN_SK	SK	Surrogate key for the user that last modified the fiscal calendar
UPD_TS	DateTime	Timestamp of the last change to this fiscal calendar record or any of its periods

FSC_CAL_DET

The FSC_CAL_DET table contains the fiscal periods for fiscal calendars defined in FSC_CAL. Fiscal calendars explicitly enumerate each period so there will be a finite number of periods defined. It is possible to encounter a nominal date that falls outside the range of fiscal period records if the fiscal calendars are not maintained or a nominal date is used farther in the future than expected.

Column Name	DataType	Description
FSC_CAL_DET_SK	SK	Surrogate key of this fiscal calendar period
FSC_CAL_SK	SK	Surrogate key of the associated fiscal calendar
LABEL	NVARCHAR(30)	Name of the fiscal period (unique with respect to the fiscal calendar)
START_NOM_DATE	Nominal date	Earliest nominal date that falls within this fiscal period
STOP_NOM_DATE	Nominal date	Latest nominal date that falls within this fiscal period

Work Rule Plug-in COM Interfaces Reference

Work rule plug-ins are COM DLLs that run in-process with the Aspect Workforce client. The Aspect Workforce client hosts instances of the work rule plug-in during automatic assignment of trial schedules and during preference-based scheduling.

Please be aware that the work rule plug-ins are called from the scheduling algorithm's inner loops so performance is very important.

ISegment Interface

The ISegment interface represents a single segment in a schedule. It provides properties that are common to general segments and detail segments. Objects that implement ISegment are read-only input to the work rule plug-in.

```
interface ISegment: IDispatch
       {
  [propget, id(0x000000C9)]
  HRESULT _stdcall NominalDate([out, retval] long* Value);
  [propget, id(0x000000CA)]
        HRESULT _stdcall SegCodeKey([out, retval] double* Value);
        [propget, id(0x000000CB)]
        HRESULT _stdcall SegmentKey([out, retval] double* Value);
        [propget, id(0x000000CC)]
        HRESULT _stdcall ParentDaySchedule([out, retval]
       IDaySchedule** Value);
};
```

ISegment Properties

Table 2: ISegment Properties

Property Name	Data Type	Description
NominalDate	Nominal Date	The integer identifier for a nominal date. Nominal dates are the number of days since December 30, 1899, UTC.
SegCodeKey	SK	The surrogate key of a segment definition
SegmentKey	SK	The surrogate key of a general or detail segment
ParentDaySchedule	Object reference	The IDaySchedule which owns this ISegment

IDetailSegment Interface

The IDetailSegment interface represents a single detail segment in a schedule. It includes all properties from ISegment and adds start and stop moments. Objects that implement IDetailSegment are read-only input to the work rule plug-in.

```
interface IDetailSegment: ISegment
{
  [propget, id(0x0000012D)]
  HRESULT _stdcall StartMoment([out, retval] long* Value);
  [propget, id(0x0000012E)]
  HRESULT _stdcall StopMoment([out, retval] long* Value);
};
```

IDetailSegment Properties

Table 3: IDetailSegment Properties

Property Name	Data Type	Description
StartMoment	Moment	The moment in time this detail segment begins as the integer number of minutes since 12:00 AM December 30, 1899, UTC
StopMoment	Moment	The moment in time this detail segment ends as the integer number of minutes since 12:00 AM December 30, 1899, UTC

IGeneralSegment Interface

The IGeneralSegment interface represents a single general segment in a schedule. It includes all properties from ISegment and does not introduce new properties of its own. Objects that implement IGeneralSegment are read-only input to the work rule plug-in.

```
interface IGeneralSegment: ISegment
{
```

IDaySchedule Interface

The IDaySchedule interface represents a collection of detail and general segments that fall on a single nominal date. Work rules evaluate a set of IDaySchedule objects and are guaranteed that only one IDaySchedule may be present for each nominal date. It is possible that a scheduling algorithm may evaluate a schedule on a nominal date when there is already an official schedule for that nominal date. In that case, the work rule should treat the entire official schedule as replaced by the candidate schedule. Objects that implement IDaySchedule are read-only input to the work rule plug-in.

```
interface IDaySchedule: IDispatch
{
  [propget, id(0x000000C9)]
  HRESULT _stdcall NominalDate([out, retval] long* Value);
  [propget, id(0x000000CA)]
  HRESULT _stdcall DetailSegmentCount([out, retval]
  long* Value);
  [propget, id(0x000000CB)]
  HRESULT _stdcall DetailSegment([in] long Index,
  [out, retval] IDetailSegment** Value);
  [propget, id(0x000000CC)]
  HRESULT _stdcall GeneralSegmentCount([out, retval]
   long* Value);
  [propget, id(0x000000CD)]
  HRESULT _stdcall GeneralSegment([in] long Index,
 [out, retval] IGeneralSegment** Value);
  [propget, id(0x000000CE)]
  HRESULT _stdcall StartMoment([out, retval] long* Value);
  [propget, id(0x000000CF)]
  HRESULT _stdcall StopMoment([out, retval] long* Value);
  [propget, id(0x000000D0)]
  HRESULT _stdcall ParentDayScheduleSet([out, retval]
   IDayScheduleSet** Value);
  [propget, id(0x000000D1)]
  HRESULT _stdcall DayScheduleID([out, retval] double* Value);
};
```

IDaySchedule Properties

Table 4: IDaySchedule Properties

Property Name	Data Type	Description
NominalDate	Nominal date	This is the integer identifier for a nominal date. The collection of detail and general segments with a common nominal date comprises the single-day schedule for that date. Nominal dates are the number of days since December 30, 1899, UTC.
DetailSegmentCount	Long integer	The number of detail segments in this single-day schedule
GeneralSegmentCount	Long integer	The number of general segments in this single-day schedule
StartMoment	Moment	This is the moment in time of the earliest detail segment start for the schedule. If there are no detail segments in the schedule, this property returns TMOMENT_MAX (500000000)
StopMoment	Moment	This is the moment in time of the latest detail segment end for the schedule. If there are no detail segments in the schedule, this property returns -TMOMENT_MAX (-500000000)
ParentDayScheduleSet	IDayScheduleSet reference	If this single-day schedule is a member of a one-day schedule set, this property returns a reference to the IDayScheduleSet. If not, this property returns NULL.
DayScheduleID	SK	Surrogate key for the single-day schedule.

IDaySchedule DetailSegment Method

```
[propget, id(0x000000CB)]
HRESULT _stdcall DetailSegment([in] long Index,
[out, retval] IDetailSegment** Value);
```

Given a zero-based index, returns the detail segment at that index.

Returns NULL if the index is out of bounds.

Table 5: IDaySchedule DetailSegment Parameters

Parameter	Description
[in] long Index	Zero-based index into the list of detail segments
[out, retval] IDetailSegment** Value	Reference to IDetailSegment associated with the given index, or NULL if the index is out of bounds

IDaySchedule GeneralSegment Method

```
[propget, id(0x000000CD)]
HRESULT _stdcall GeneralSegment([in] long Index,
   [out, retval] IGeneralSegment** Value);
```

Given a zero-based index, returns the general segment at that index.

Returns NULL if the index is out of bounds.

Table 6: IDaySchedule GeneralSegment Parameters

Parameter	Description
[in] long Index	Zero-based index into the list of general segments
[out, retval] IGeneralSegment** Value	Reference to the IGeneralSegment associated with the given index, or NULL if the index is out of bounds

IDayScheduleSet Interface

This represents a set of one or more single-day schedules that may be assigned to an employee as a unit. There may be nominal dates within the set that do not have a IDaySchedule. The work rule should treat these days as either undefined, meaning future scheduling runs may or may not assign a schedule on that date, or rely on an existing official schedule if one exists for the missing nominal date.

As the scheduling algorithm evaluates a set of candidate schedules, IDayScheduleSet is the level that work rules are asked to append to their working set of schedules. The algorithm will also remove IDayScheduleSet objects as alternatives are considered. Objects that implement IDayScheduleSet are read-only input to the work rule.

```
interface IDayScheduleSet: IDispatch
{
  [propget, id(0x000000C9)]
```

```
  HRESULT _stdcall DayScheduleCount([out, retval] long* Value);
  [propget, id(0x000000CA)]
  HRESULT _stdcall DaySchedule([in] long Index,
[out, retval] IDaySchedule** Value);
  [propget, id(0x000000CB)]
  HRESULT _stdcall StartNominalDate([out, retval] long* Value);
  [propget, id(0x000000CC)]
  HRESULT _stdcall StopNominalDate([out, retval] long* Value);
  [propget, id(0x000000CD)]
  HRESULT _stdcall DayScheduleByNominalDate([in] long NominalDate,
  [out, retval] IDaySchedule** Value);
  [propget, id(0x000000CE)]
  HRESULT _stdcall DayScheduleSetID([out, retval] double* Value);
  [propget, id(0x000000CF)]
  HRESULT _stdcall ParentDayScheduleSetList(
  [out, retval] IDayScheduleSetList** Value);
};
```

Table 7: IDayScheduleSet Properties

Property Name	Data Type	Description
DayScheduleCount	Long integer	This is the number of single-day schedules within this IDayScheduleSet.
StartNominalDate	Nominal date	This is the earliest nominal date for all single-day schedules in the IDayScheduleSet. If there are no schedules in this IDayScheduleSet, this property is undefined.
StopNominalDate	Nominal date	This is the latest nominal date for all single-day schedules in the IDayScheduleSet. If there are no schedules in this IDayScheduleSet, this property is undefined.
DayScheduleSetID	SK	Surrogate key for the IDayScheduleSet object
ParentDayScheduleSetList	IDayScheduleSetList *	If this IDayScheduleSet is contained in an IDayScheduleSetList, this returns a reference to the container.

		If this IDayScheduleSet is not contained in an IDayScheduleSetList, this property returns NULL.

IDayScheduleSet DaySchedule Method

```
[propget, id(0x000000CA)]
HRESULT _stdcall DaySchedule([in] long Index,
[out, retval] IDaySchedule** Value);
```

Given a zero-based index, returns the IDaySchedule at that index.

Returns NULL if either Index is out of bounds or there is no IDaySchedule at the given index.

Table 8: IDayScheduleSet DaySchedule Parameters

Parameter	Description
[in] long Index	Zero-based index into the list of IDaySchedules
[out, retval] IDaySchedule** Value);	Reference to the IDaySchedule associated with the given index, or NULL if the index is out of bounds

IDayScheduleSet DayScheduleByNominalDate Method

```
[propget, id(0x000000CD)]
HRESULT _stdcall DayScheduleByNominalDate([in] long NominalDate,
[out, retval] IDaySchedule** Value);
```

Given a nominal date, returns an IDaySchedule reference if one exists for the given nominal date.

Returns NULL if there is no IDaySchedule in the set on the given nominal date. The work rule should treat the day as undefined (as if it could receive any schedule in the future).

Table 9: IDayScheduleSet DayScheduleByNominalDate Parameters

Parameter	Description
[in] long NominalDate	Nominal date of the IDaySchedule we wish to access
[out, retval] IDaySchedule** Value	Reference to the IDaySchedule for the given nominal date, or NULL there is no IDaySchedule with the given nominal date in the IDayScheduleSet

IDayScheduleSetList Interface

The IDayScheduleSetList interface represents a list of candidate schedules. After a scheduling algorithm has generated candidate schedules for evaluation, it first notifies the work rules of all candidate schedules so that the work rules can preprocess each candidate schedule and cache computed schedule information required for evaluation. This is called setting the Schedule Scope, which will be discussed in more detail later. The work rule may need to evaluate many combinations of these schedule sets as the algorithm searches for optimal and allowable schedule assignments. Preprocessing the IDayScheduleSetList should reduce the performance impact of combinatorial explosion as many schedule combinations are evaluated.

Objects that implement IDayScheduleSetList are read-only input to the work rule.

```
interface IDayScheduleSetList: IDispatch
{
  [propget, id(0x000000C9)]
  HRESULT _stdcall DayScheduleSetCount([out, retval] long* Value);
  [propget, id(0x000000CA)]
  HRESULT _stdcall DayScheduleSet([in] long Index,
   [out, retval] IDayScheduleSet** Value);
};
```

Table 10: IDayScheduleSetList Properties

Property Name	Data Type	Description
DayScheduleSetCount	Long integer	The number of schedule sets in this IDayScheduleSetList

IDaySCheduleSetList DayScheduleSet Method

```
[propget, id(0x000000CA)]
HRESULT _stdcall DayScheduleSet([in] long Index,
 [out, retval] IDayScheduleSet** Value);
```

Given a zero-based index, returns the IDayScheduleSet at that index.

If the index is out of bounds, returns NULL.

Table 11: IDayScheduleSetList DayScheduleSet Parameters

Parameter	Description
[in] long Index	Zero-based index into the list of IDayScheduleSet
[out, retval] IDayScheduleSet** Value	Reference to the IDayScheduleSet associated with the given index, or NULL if the index is out of bounds

IWorkRule Interface

The work rule interface is the interface that a work rule plug-in must implement. A work rule plug-in is given an employee context, a scheduling context, and then is repeatedly asked to evaluate schedules, add schedules to the candidate solution, and remove schedules from the candidate solution as the scheduling algorithm searches for a scheduling solution that balances staff requirements, employee preferences, and the probability that work rules will be satisfied.

```
interface IWorkRule: IDispatch
{
  [id(0x000000C9)]
  HRESULT _stdcall GetCredentials([in] long LangID,
  [in] BSTR DBAlias,
  [in, out] BSTR* Credentials);
```

```
  [id(0x000000CA)]
  HRESULT _stdcall GetConfigurationInfo([in] long LangID,
[in] BSTR DBAlias,
[in] BSTR Credentials,
[in, out] BSTR* ConfigInfo,
[in, out] BSTR* DisplayText,
[in] VARIANT_BOOL ShowDialog);

[id(0x000000CB)]
  HRESULT _stdcall InitializeWorkRule([in] long LangID,
 [in] BSTR DBAlias,
 [in] BSTR Credentials,
 [in] BSTR ConfigInfo,
 [in, out] VARIANT_BOOL* ValidConfigInfo);
  [id(0x000000CC)]
  HRESULT _stdcall FinalizeWorkRule(void);
  [id(0x000000CD)]
  HRESULT _stdcall BeginScheduleScope([in] IDayScheduleSetList*
DayScheduleSetList,
  [in] long StartNominalDate,
  [in] long StopNominalDate);
  [id(0x000000CE)]
  HRESULT _stdcall EndScheduleScope(void);
  [id(0x000000CF)]
  HRESULT _stdcall BeginEmployeeScope([in] double EmployeeKey);
  [id(0x000000D0)]
  HRESULT _stdcall EndEmployeeScope(void);
  [id(0x000000D1)]
  HRESULT _stdcall EvaluateDayScheduleSet([in] IDayScheduleSet*
   DayScheduleSet,
 [out] VARIANT_BOOL* Pass,
 [out] double* Score);
  [id(0x000000D2)]
  HRESULT _stdcall AddDayScheduleSet([in] IDayScheduleSet*
  DayScheduleSet);
  [id(0x000000D3)]
  HRESULT _stdcall RemoveDayScheduleSet([in] IDayScheduleSet*
 DayScheduleSet);
};
```

IWorkRule GetCredentials Method

```
[id(0x000000C9)]
HRESULT _stdcall GetCredentials([in] long LangID,
[in] BSTR DBAlias,
[in, out] BSTR* Credentials);
```

The GetCredentials method should present a user interface to request security credentials from the user, encode the credentials into a single string, then encrypt and return the encrypted string. These encrypted credentials will be passed back into the work rule plug-in as a parameter to the InitializeWorkRule method. It is up to the work rule plug-in to define its internal encryption-decryption scheme.

Note: This method is not currently called by Aspect Workforce. In the future, there will be a utility or module to facilitate registering third-party work rule plug-ins. For now, the security credentials must be collected, encrypted, and stored in the RULE_PLUG_IN table using custom tools or SQL scripts.

Work rule plug-ins may ignore the credentials and attempt to use Integrated Security when logging into the Aspect Workforce database. In this case, the work rule plug-in will inherit the credentials of the Windows user running the Aspect Workforce client.

Table 12: IWorkRule GetCredentials Parameters

Parameter	Description
[in] long LangID	Windows primary and secondary locale ID.
[in] BSTR DBAlias	The identifier for the Aspect Workforce database instance.
[in, out] BSTR* Credentials	These are the encrypted credentials that the work rule requires to access resources such as the Aspect Workforce database or web services. These credentials will be passed to the work rule as a parameter to InitializeWorkRule.

IWorkRule GetConfigurationInfo Method

```
[id(0x000000CA)]
HRESULT _stdcall GetConfigurationInfo([in] long LangID,
[in] BSTR DBAlias,
[in] BSTR Credentials,
[in, out] BSTR* ConfigInfo,
[in, out] BSTR* DisplayText,
[in] VARIANT_BOOL ShowDialog);
```

The GetConfigurationInfo method returns configuration for a work rule implemented using this work rule plug-in. If ShowDialog is TRUE, the work rule plug-in should display an application modal dialog to allow the user to set or edit work rule configuration. If ShowDialog is FALSE, the work rule should return the default configuration.

GetConfiguration is called from the Aspect Workforce client Work Rules module when the user clicks Configure for a work rule, or clicks OK without configuring the work rule.

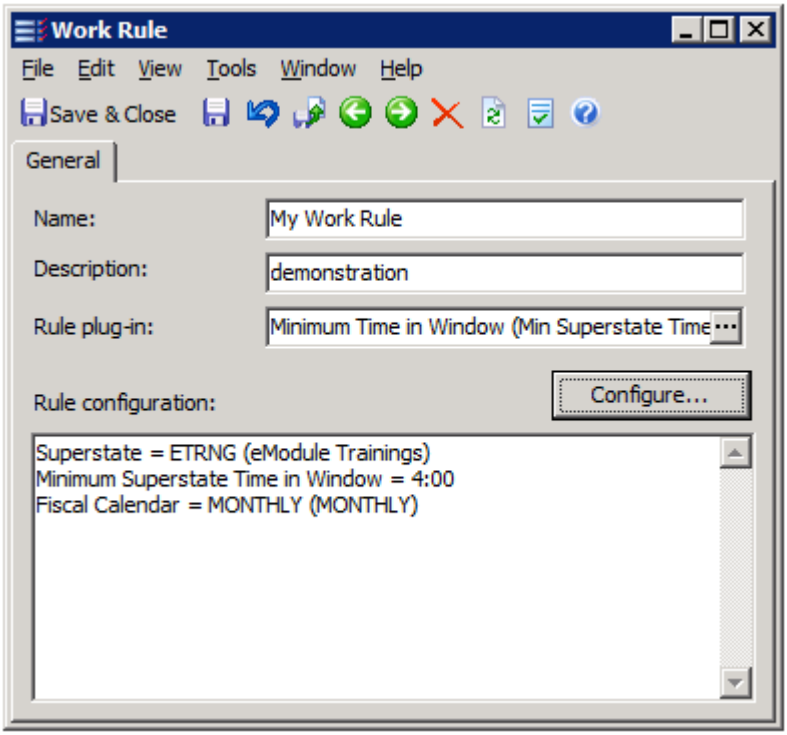


Figure 5: Work Rule Editor

Table 13: IWorkRule GetConfigurationInfo Parameters

Parameter	Description
[in] long LangID	Windows primary and secondary locale ID
[in] BSTR DBAlias	Identifier for the Aspect Workforce database instance
[in] BSTR Credentials	This is the encrypted credentials for the work rule plug-in. It is the responsibility of the work rule plug-in to decrypt and use these credentials.
[in, out] BSTR* ConfigInfo	String-encoded configuration information for the work rule plug-in. If ShowDialog is TRUE and an input string is present, the work rule plug-in should allow editing of configuration; otherwise the plug-in should use default configuration and allow the user to configure a new work rule if ShowDialog is FALSE, the work rule plug-in should simply return the default configuration in a string-encoded format such as XML or JSON.
[in, out] BSTR* DisplayText	Display text that is presented to the user as a summary of the current work rule configuration (see the figure above).
[in] VARIANT_BOOL ShowDialog	This is a flag that indicates whether the work rule plug-in should present an application modal dialog to allow the user to configure a work rule (TRUE), or

Parameter	Description
	simply return the default configuration without displaying a dialog (FALSE).

IWorkRule InitializeWorkRule Method

```
[id(0x000000CB)]
HRESULT _stdcall InitializeWorkRule([in] long LangID,
[in] BSTR DBAlias,
[in] BSTR Credentials,
[in] BSTR ConfigInfo,
[in, out] VARIANT_BOOL* ValidConfigInfo);
```

The InitializeWorkRule method is called at the beginning of the scheduling algorithm to allow the work rule instance to set its internal state to reflect the Aspect Workforce database, credentials, and work rule configuration. The work rule should read or precompute any data that will be static over the scheduling run. The work rule should validate the configuration and return TRUE in ValidConfiguration if the configuration is valid and FALSE otherwise.

Table 14: IWorkRule InitializeWorkRule Parameters

Parameter	Description
[in] long LangID	Windows primary and secondary locale ID
[in] BSTR DBAlias	Identifier for the Aspect Workforce database instance
[in] BSTR Credentials	This is the encrypted credentials for the work rule plug-in. It is the responsibility of the work rule plug-in to decrypt and use these credentials.
[in] BSTR ConfigInfo	String-encoded configuration information for the work rule plug-in.
[in, out] VARIANT_BOOL* ValidConfigInfo	The work rule plug-in will validate the provided configuration and return whether the provided configuration information was valid (TRUE) or not (FALSE). If the work rule returns FALSE in this field, the scheduling run will stop with an error message.

IWorkRule FinalizeWorkRule Method

```
[id(0x000000CC)]
HRESULT _stdcall FinalizeWorkRule(void);
```

The FinalizeWorkRule method is called just before the work rule is released by the Aspect Workforce client. The work rule plug-in should free resources in this method and revert to an unconfigured state.

There are no parameters to this method.

IWorkRule BeginScheduleScope Method

```
[id(0x000000CD)]
HRESULT _stdcall BeginScheduleScope([in] IDayScheduleSetList*
DayScheduleSetList,
[in] long StartNominalDate,
[in] long StopNominalDate);
```

After initializing a work rule, the scheduling algorithm will notify each work rule of the set of candidate schedules that might be used during the scheduling run. This allows the work rules to preprocess the list of schedule sets to perform tasks such as classify the DOW, count superstate hours, or determine whether the schedule is working or non-working, etc. Since the work rule could be presented the same schedule set multiple times but in different combinations, for performance reasons it is important that any relevant computations that can be performed in this method are calculated and cached.

Table 15: IWorkRule BeginScheduleScope Parameters

Parameter	Description
[in] IDayScheduleSetList* DayScheduleSetList	The list of candidate IDayScheduleSet that may be assigned this scheduling run
[in] long StartNominalDate	This is the earliest nominal date in the scheduling run. The work rule may need to read official schedules from earlier nominal dates in order to fully evaluate its rule logic.
[in] long StopNominalDate	This is the latest nominal date in the scheduling run. The work rule may need to read official schedules from later nominal dates in order to fully evaluate its rule logic.

IWorkRule EndScheduleScope Method

```
[id(0x000000CE)]
HRESULT _stdcall EndScheduleScope(void);
```

The EndScheduleScope method is called at the end of scheduling for one schedule scope. The work rule should discard its cache of schedules and their computed properties. The scheduling algorithm may begin a new schedule scope with a new set of schedules to evaluate.

There are no parameters to this method.

IWorkRule BeginEmployeeScope Method

```
[id(0x000000CF)]
HRESULT _stdcall BeginEmployeeScope([in] double EmployeeKey);
```

After the work rule has precomputed properties of the schedules in the schedule scope, the scheduling algorithm will then assign schedules one employee at a time. Before assigning schedules to an employee, all work rules are notified that a new employee scope is beginning. The work rule should preload any employee-specific information it requires, such as employee time zone.

Table 16: IWorkRule BeginEmployeeScope Parameters

Parameter	Description
[in] double EmployeeKey	The surrogate key of the employee about to be assigned schedules

IWorkRule EndEmployeeScope Method

```
HRESULT _stdcall EndEmployeeScope(void);
[id(0x000000D1)]
```

When the scheduling algorithm is finished assigning schedules to an employee, it calls EndEmployeeScope for all work rules. The work rule should discard any cached information associated with the employee as well as the set of schedules that are cached in its active list of IDayScheduleSet objects. The scheduling algorithm may begin a new employee scope.

There are no parameters to this method.

IWorkRule EvaluateDayScheduleSet Method

```
[id(0x000000D1)]
HRESULT _stdcall EvaluateDayScheduleSet([in] IDayScheduleSet*
 DayScheduleSet,
[out] VARIANT_BOOL* Pass,
[out] double* Score);
```

The EvaluateDayScheduleSet method takes into the given DayScheduleSet, any IDayScheduleSets added to the context using AddDayScheduleSet (described in the next section), and any non-overlapping official schedules to determine whether the rule is satisfied or not, and the probability that future schedule assignments will cause the rule be satisfied or not. Please the section 0 on work rule scoring for more information on calculating a work rule score.

Table 17: IWorkRule EvaluateDayScheduleSet Parameters

Parameter	Description
[in] IDayScheduleSet* DayScheduleSet	The candidate IdayScheduleSet to evaluate given the current context of official schedules and assigned candidate schedules
[out] VARIANT_BOOL* Pass	This is a Boolean flag to indicate whether the work rule is currently satisfied or not. The work rule should return TRUE if the rule is currently satisfied. The work rule should return FALSE if it is not currently satisfied.
[out] double* Score	This is the probability that future schedule assignments may cause the rule to be satisfied. 1 - Score is the probability that future schedule assignments will cause the work rule to fail.

IWorkRule AddDayScheduleSet Method

```
[id(0x000000D2)]
```

```
HRESULT _stdcall AddDayScheduleSet([in] IDayScheduleSet*
DayScheduleSet);
```

The scheduling algorithm will evaluate many combinations of schedule assignments. Each piece of the scheduling assignment will be evaluated incrementally, and if it is satisfactory, AddDayScheduleSet will be called to cause the work rule to add that IDayScheduleSet to its current evaluation context. The work rule should precompute what it can about the new evaluation context.

Table 18: IWorkRule AddDayScheduleSet Parameters

Parameter	Description
[in] IDayScheduleSet* DayScheduleSet	The candidate IdayScheduleSet to add to the current evaluation context

IWorkRule RemoveDayScheduleSet Method

```
[id(0x000000D3)]
HRESULT _stdcall RemoveDayScheduleSet([in] IDayScheduleSet*
DayScheduleSet);
```

As the scheduling algorithm backtracks in its search for a good schedule assignment, it will remove IDayScheduleSet assignments from the work rule's evaluation context. The work rule should precompute what it can about the new evaluation context.

Table 19: IWorkRule RemoveDayScheduleSet Parameters

Parameter	Description
[in] IDayScheduleSet* DayScheduleSet	The candidate IdayScheduleSet to remove from the current evaluation context

Work Rule Scheduling Scoring

Work rules provide a two-part evaluation result

- A Boolean flag named Pass indicating if the rule is currently satisfied
- A numeric Score between zero and one indicating the probability future assignments will satisfy the rule

When the Score is zero, it indicates that if the schedule being evaluated were assigned to the employee – given the current evaluation context – it would be in violation of the work rule.

When the Score is between zero and one it means that there is uncertainty and the rule could change the value Pass depending on future schedule assignments.

When the Score is one, it indicates that the rule is satisfied and no future schedule assignments will violate the work rule in the given window of nominal dates.

Work Rule Scoring Example

In this example, we will consider a hypothetical work rule that limits schedules to at most three consecutive long shifts of 12 hours or more. To make this decision, the work rule needs to differentiate nominal dates with a shift of 12 hours or more from those with less than 12 hours, including zero.

Assume the scheduling run is for one work week Monday November 3 through Friday November 7. Since the work rule needs to limit schedules based on a 3-day sliding window, it will need the official schedules from Saturday November 1 through Sunday November 9.

Let's assume we start out with the following official schedules in this range.

Yes=12 hours or more		?=Unknown						
November								
Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	2	3	4	5	6	7	8	9
Yes	Yes	**?**	**?**	**?**	**?**	**?**	?	Yes

As far as we can see, we have not violated the three consecutive long days rule.

Now, let's assume we have 4 12-hour shifts we need to place and we want to evaluate the best days we could place these shifts within our schedule.

Let's examine the first day one option at a time.

November								
Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	2	3	4	5	6	7	8	9
Yes	Yes	**Yes**	**?**	**?**	**?**	**?**	?	Yes

If we place a 12-hour shift on Monday, it completes a run of 3 consecutive 12-hour shifts. We have 5 days in our window of dates that are uncertain. They might yet receive a 12-hour shift, and they might not. With no additional knowledge, we will assume equal probability of getting a 12-hour shift and not getting a 12-hour shift.

Here is a table of all possibilities and their Pass / Fail outcomes after adding a 12-hour shift to Monday.

November									
Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun	
1	2	3	4	5	6	7	8	9	Pass/Fail
Yes	Yes	**Yes**						**Yes**	Pass
Yes	Yes	**Yes**	**Yes**					**Yes**	Fail
Yes	Yes	**Yes**		**Yes**				**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**				**Yes**	Fail
Yes	Yes	**Yes**			**Yes**			**Yes**	Pass
Yes	Yes	**Yes**	**Yes**		**Yes**			**Yes**	Fail
Yes	Yes	**Yes**		**Yes**	**Yes**			**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**	**Yes**			**Yes**	Fail
Yes	Yes	**Yes**				**Yes**		**Yes**	Pass
Yes	Yes	**Yes**	**Yes**			**Yes**		**Yes**	Fail
Yes	Yes	**Yes**		**Yes**		**Yes**		**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**		**Yes**		**Yes**	Fail
Yes	Yes	**Yes**			**Yes**	**Yes**		**Yes**	Pass
Yes	Yes	**Yes**	**Yes**		**Yes**	**Yes**		**Yes**	Fail
Yes	Yes	**Yes**		**Yes**	**Yes**	**Yes**		**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**		**Yes**	Fail
Yes	Yes	**Yes**					**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**				**Yes**	**Yes**	Fail
Yes	Yes	**Yes**		**Yes**			**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**			**Yes**	**Yes**	Fail
Yes	Yes	**Yes**			**Yes**		**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**		**Yes**		**Yes**	**Yes**	Fail
Yes	Yes	**Yes**		**Yes**	**Yes**		**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**	**Yes**		**Yes**	**Yes**	Fail
Yes	Yes	**Yes**				**Yes**	**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**			**Yes**	**Yes**	**Yes**	Fail
Yes	Yes	**Yes**		**Yes**		**Yes**	**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**		**Yes**	**Yes**	**Yes**	Fail
Yes	Yes	**Yes**			**Yes**	**Yes**	**Yes**	**Yes**	Fail
Yes	Yes	**Yes**	**Yes**		**Yes**	**Yes**	**Yes**	**Yes**	Fail
Yes	Yes	**Yes**		**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	Fail
Yes	Yes	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	Fail

Figure 6: Work Rule Evaluation Example 1

If we count up all pass and fail outcomes after adding a 12-hour shift on Monday, we have

Pass Count = 14

Fail Count = 18

So adding a Monday 12-hour hour shift will evaluate to:

Pass = TRUE

Score = 14 / 32 = 0.4375 or a 43.75% chance the rule will pass future assignments.

There is a 56.25% chance future assignments will fail.

In comparison, let's consider placing the first 12-hour shift on Wednesday.

November									
Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun	
1	2	3	4	5	6	7	8	9	Pass/Fail
Yes	Yes			**Yes**				**Yes**	Pass
Yes	Yes	**Yes**		**Yes**				**Yes**	Pass
Yes	Yes		**Yes**	**Yes**				**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**				**Yes**	Fail
Yes	Yes			**Yes**	**Yes**			**Yes**	Pass
Yes	Yes	**Yes**		**Yes**	**Yes**			**Yes**	Pass
Yes	Yes		**Yes**	**Yes**	**Yes**			**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**	**Yes**			**Yes**	Fail
Yes	Yes			**Yes**		**Yes**		**Yes**	Pass
Yes	Yes	**Yes**		**Yes**		**Yes**		**Yes**	Pass
Yes	Yes		**Yes**	**Yes**		**Yes**		**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**		**Yes**		**Yes**	Fail
Yes	Yes			**Yes**	**Yes**	**Yes**		**Yes**	Pass
Yes	Yes	**Yes**		**Yes**	**Yes**	**Yes**		**Yes**	Pass
Yes	Yes		**Yes**	**Yes**	**Yes**	**Yes**		**Yes**	Fail
Yes	Yes	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**		**Yes**	Fail
Yes	Yes			**Yes**			**Yes**	**Yes**	Pass
Yes	Yes	**Yes**		**Yes**			**Yes**	**Yes**	Pass
Yes	Yes		**Yes**	**Yes**			**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**			**Yes**	**Yes**	Fail
Yes	Yes			**Yes**	**Yes**		**Yes**	**Yes**	Pass
Yes	Yes	**Yes**		**Yes**	**Yes**		**Yes**	**Yes**	Pass
Yes	Yes		**Yes**	**Yes**	**Yes**		**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**	**Yes**		**Yes**	**Yes**	Fail
Yes	Yes			**Yes**		**Yes**	**Yes**	**Yes**	Pass
Yes	Yes	**Yes**		**Yes**		**Yes**	**Yes**	**Yes**	Pass
Yes	Yes		**Yes**	**Yes**		**Yes**	**Yes**	**Yes**	Pass
Yes	Yes	**Yes**	**Yes**	**Yes**		**Yes**	**Yes**	**Yes**	Fail
Yes	Yes			**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	Fail
Yes	Yes	**Yes**		**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	Fail
Yes	Yes		**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	Fail
Yes	Yes	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	Fail

Figure 7: Work Rule Evaluation Example 2

Pass Count = 20

Fail Count = 12

So adding a Monday 12-hour hour shift will evaluate to:

Pass = TRUE

Score = 20 / 32 = 0.625 or a 62.5% chance the rule will pass future assignments.

There is a 37.5% chance future assignments will fail.

It turns out that Wednesday or Thursday are the best days to start placing schedules based solely on this rule. We are able to place all 4-12 hour schedules without exceeding 3 consecutive with this solution:

November								
Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	2	3	4	5	6	7	8	9
Yes	Yes	**Yes**		**Yes**	**Yes**	**Yes**		Yes

As far as the work rule is concerned this would evaluate to:

Pass = TRUE

Score = 0.25.

Remember we are assuming an equal probability that any single day will get a 12-hour shift in the future or not.

It is important that work rules are efficient and precalculate as much of this as possible since evaluation happens in the innermost schedule loops.

There are many techniques to calculating probabilities, and explicit enumeration of the entire state space works adequately, but can be inefficient. Random sampling of large state spaces can yield good results, but care should be taken when there are unique or very small numbers of allowed solutions. Random sampling might not find any and prevent allowed assignments.

Pass / Fail Work Rules

If probabilities are difficult or impossible to calculate for a work rule, it is possible to implement a Pass / Fail rule that returns a Score of one (1) whenever the rule passes, and zero (0) whenever the rule fails. Such a rule will prevent the assignment of a schedule that violates the rule, but does not provide any guidance to the scheduling algorithm. Without guidance from the rule, the scheduling algorithm could easily make early decisions that later prevent schedule assignments.

Work Rule Plug-in Implementation Steps

This section provides an outline of the tasks necessary to implement a custom work rule plug-in.

1. Identify a work rule that cannot be implemented or sufficiently approximated using the standard Aspect Workforce work rule types. The work rule description should answer the question, “What schedule assignments will the rule prevent?” Ultimately, preventing schedule assignments that would violate the work rule is all work rule plug-ins can do. They do not generate schedules and they cannot alter the behavior of any other work rules or built-in scheduling constraints.

2. Identify parameters to the work rule that will allow the work rule plug-in to implement a family of related work rules. Most work rules have parameters such as the size of a date window, allowing the user to select a fiscal calendar, or specifying some number of superstate hours.

3. Decide if your work rule will calculate probability of passing future schedule assignments or if your work rule will simply implement Pass / Fail behavior. If your work rule will calculate probabilities, you need to design an algorithm to perform the probability calculation and test it outside of the work rule context to ensure correctness.

4. Decide if your work rule requires credentials to access the Aspect Workforce database, web services, or other resources. Your work rule plug-in may instead rely on Integrated Security and inherit the credentials of the user running the Aspect Workforce client. If your work rule plug-in requires its own credentials, you will need to create and encrypt a string that encodes these credentials. At this time, there is no facility to call IWorkRule.GetCredentials.

5. Select a programming language to implement your custom work rule plug-in and create a 64-bit COM DLL project.

6. Import the WorkRuleInterface.tlb type library into your project.

7. Create a public COM class that implements the IWorkRule interface.

8. Design your user interface for editing the work rule configuration.

9. Implement each method of your work rule plug-in to adhere to the interface behavior as described in this document.

10. Register your work rule plug-in with Aspect Workforce by adding a record into the RULE_PLUG_IN table in the Aspect Workforce database.

11. Install and register your plug-in DLL on each machine where the Aspect Workforce client will be used for managing work rules, work rule sets, and scheduling tasks.

12. Test your work rule plug-in against a wide variety of trial and official schedules.

Performance Guidelines

Work rules are called upon to evaluate combinations of schedules as the scheduling algorithms are searching for good schedule assignments. The scheduling algorithms do not only iterate over employees and schedules, but over many *combinations* of schedules. Work rules plug into the innermost loops of these algorithms and poor work rule plug-in performance will significantly impact the time required for scheduling runs. It is extremely important that work rules are as efficient as possible, especially for the routines AddDayScheduleSet, RemoveDayScheduleSet, and EvaluateDayScheduleSet. These routines should rely on cached information as much as possible and avoid accessing the Aspect Workforce database or web services.

Work Rule Plug-in Integration

Installing a Work Rule Plug-in involves

- a COM DLL that provides the executable code for work rule plug-in. This may be implemented in most languages supporting COM interfaces and graphical user interfaces. The DLL must be installed and registered on all computers where the Aspect Workforce client will be used to perform scheduling tasks.
- a database record in the RULE_PLUG_IN table that provides the information required to allow a user to select the work rule plug-in, the ProgId that allows launching the plug-in, and a description of the security model for the plug-in.

Work Rule Plug-in Registration

To register a new work rule plug-in with Aspect Workforce, a single record needs to be added to the RULE_PLUG_IN table.

Example Script for SQL Server

```
USE [WFM_DB];
GO


-- if it is necessary to set the application role uncomment these lines and
replace the default password (tcs_cli3nt) with the correct one
--DECLARE @cookie varbinary(8000);
--EXEC sp_setapprole 'TCS_CLIENT', 'tcs_cli3nt', @fCreateCookie = true, @cookie
= @cookie OUTPUT;
BEGIN TRAN T1;


DECLARE @RC int;
DECLARE @sk_out numeric(15,1)


EXECUTE @RC = [GET_SK] @sk_out OUTPUT;


INSERT INTO [RULE_PLUG_IN]
   ([RULE_PLUG_IN_SK]
   ,[NAME]
   ,[DESCR]
   ,[SECUR_TYPE]
   ,[PROG_ID]
   ,[SECUR_BLOB])
 VALUES
   (@sk_out
   ,'My Custom Work Rule Plug-in'
   ,'Implements custom rules'
   ,'P'
   ,'MyCompany.MyFirstWorkRule'
   ,'brNo5TFjDttOkYoFXSqPHv1tBdvqI3mu/WFgoWMq');


COMMIT TRAN T1;
--EXEC sp_unsetapprole @cookie;
GO
```

Example Script for Oracle

```
set role tcs_client identified by tcs_client;

variable v_sk NUMBER;
execute EWFM80.GET_SK (:v_sk);

INSERT INTO EWFM80.RULE_PLUG_IN
   (RULE_PLUG_IN_SK
   ,NAME
   ,DESCR
   ,SECUR_TYPE
   ,PROG_ID
   ,SECUR_BLOB)
 VALUES
   (:v_sk
   ,'My Custom Work Rule Plug-in'
   ,'Implements custom rules'
   ,'P'
   ,'MyCompany.MyFirstWorkRule'
   ,'brNo5TFjDttOkYoFXSqPHv1tBdvqI3mu/WFgoWMq');

commit;
```

Work Rule Plug-in Installation

Work rule plug-in COM DLLs must be installed and registered with the Windows COM system. The DLL must be deployed locally on each machine where the Aspect Workforce client will be used to execute scheduling runs. In general, COM DLLs can be registered using the Windows utility regsvr32.exe or regasm.dll for .Net assemblies. Please consult the documentation for your development language for instructions on registering COM DLLs written using your language and libraries.

Installing the work rule plug-in on the Aspect Workforce client machine typically involves these steps:

1. Perform the steps necessary for Aspect Workforce client to install and run.

2. Install any prerequisites for your custom work rule plug-in, such as a specific version of the .Net Framework, or other runtime libraries.

3. Install your custom work rule plug-in DLL where appropriate for your runtime requirements. In general, custom executables should not be installed in folders created by Aspect Workforce installers, as these folders may be overwritten during upgrades. COM DLLs written in .Net must either be installed in the same folder as TCS.exe or in the .Net GAC. Care must be taken during updates of Aspect Workforce that custom executables remain installed.

4. Register your work rule plug-in COM DLL using the appropriate 64-bit utility:
   C:\Windows\System32\regsvr32.exe
   C:\Windows\Microsoft.Net\Framework64\v4.0.30319\RegAsm.exe
   Please consult the documentation of your development tools for information on deploying 64-bit COM DLL components.

Work Rule Plug-in Example

The Aspect Workforce SDK includes an example work rule plug-in. This example is intended to be educational, and the example code should not be used in a production environment. The code has been simplified for example purposes and in particular is not hardened for error handling, security, or even correctness when examining schedules.

SampleWorkRule.csproj

A work rule project is a DLL that publishes COM classes implementing the IWorkRule COM interface. In .Net, you should create a new C# class library project. In the project Build options, you should check the option Register for COM interop to enable registering COM visible classes with Windows.

SampleWorkRule.cs

This file contains the class that implements IWorkRule and is the public interfaces for your custom work rule logic. Each work rule must have a unique Guid and ProgId.

```
[Guid("F20B9E82-A15C-48DB-BD1E-08AA1CDDE33F")]
[ProgId("Aspect.EWFM.SDK.SampleWorkRule")]
[ComVisible(true)]
public class SampleWorkRule : IWorkRule
```

SampleWorkRuleConfig.cs

Each work rule plug-in should support some configuration parameters to avoid hard coding details that users might want to customize. These configuration options should be adjustable by a configuration dialog your plug-in presents from GetConfigurationInfo. The configuration is serialized into a string for storage in the Aspect Workforce database. The sample work rule serializes credentials and configuration into JSON format.

SampleWorkRuleCredentials.cs

The example credentials are not really used by the sample. In your solution, you may opt to support credentials or use integrated security and inherit the credentials of the user running the Aspect Workforce client. If you need to support credentials, these must be encrypted before returning them from the GetCredentials function. Aspect Workforce will return the encrypted credentials to the work rule on initialization. The work rule plug-in should decrypt the credentials and deserialize them for use.

SampleWorkRuleConfigEditor

The work rule-plug-in needs to provide a dialog to allow configuring work rule instances within Aspect Workforce.

SampleDayScheduleProperties.cs

Work rules generally define some predicate on employee schedules and then impose some constraint on the number of occurrences or their pattern within an employee's schedule. The idea is to first classify each single-day schedule in the schedule scope, and cache this classification for fast reference during rule evaluation. The SampleDayScheduleProperties is a very simple example that simply remembers if a DaySchedule instance has detail segments. A real work rule might examine staff hours to determine whether an employee is working and look at the day of week to determine if the day is a holiday or weekend.

SampleDayScheduleSetProperties.cs

Schedules are assigned to employees as DayScheduleSets. The SampleDayScheduleSetProperties class caches information about the day schedule sets so when a DayScheduleSet is added to the evaluation context, the work rule can quickly look up information about that set of schedules.

SampleScheduleContext.cs

After the scheduling algorithm has generated a new set of candidate schedules, it notifies each work rule to preprocess the generated schedules and cache whatever information it needs about them. The SampleScheduleContext class keeps a dictionary of SampleDayScheduleSetProperties that are quickly accessed by DayScheduleSetID.

SampleEmployeeContext.cs

Before assigning schedules to an employee, each work rule is notified about the employee whom schedules will be assigned to. This allows the work rule to look up and cache any relevant information about the employee. Such information might include the employee time zone and official schedules within and near the scheduling date window. The example loads summary information to act as a proxy for determining if a schedule has any work segments. This technique is quick but may be inaccurate. Most work rules examine the full set of official segments and calculate staff hours, durations, and other information from that data.

SampleEvaluationContext.cs

When a work rule is asked to evaluate a DayScheduleSet, it is not only evaluating the given set of schedules, but evaluating them in the context of official schedules and any other schedules that were added to the evaluation context through calls to AddDayScheduleSet. The SampleEvaluationContext is a simple example that just remembers if each nominal date has a matching official schedule or trial schedule. AddDayScheduleSet adds to the set of trial schedules and RemoveDayScheduleSet removes schedules from the set of trial schedules. The set of official schedules are static for an employee context, but official schedules may be overridden if a trial schedule appears on the same nominal date.

SampleScoreCalculator.cs

The score for a work rule is the probability that the work rule will approve the schedules in the current evaluation context and any possible additional schedule assignments for future nominal dates. The example is an implementation of a Pass / Fail work rule which returns a score of 1.0 for an evaluation context that is acceptable and 0.0 for any evaluation context that violates the rule. This technique is fast and easy to show as correct, but does not provide guidance to the scheduling algorithm for which candidate schedules might be preferred.

Using the Web Services

This section describes the Aspect Workforce web services that are provided with the SDK.

Prerequisites

Table 1 shows the requirements for using the web services.

Table 1: Aspect Workforce Requirements

Versions	Aspect Workforce 7.5 or later
Enhancement Packages	None additional

Launching the Online Help

The SDK includes a Help system that documents a large selection of web services in Aspect Workforce. For each web service, the Help provides the following documentation:

- Schema
- Input parameters
- Positive acknowledgment
- Negative acknowledgment
- Functional requirements
- XML request syntax
- XML response syntax

To launch the Help:

1. Browse to the folder where the Help system is installed. The default path is:

```
C:\Program Files (x86)\
Alvaria\Workforce 25.0 SDK\Web_Services_Related_Files
```

2. Open the folder, and double-click the file WFM250_WebServices_Help.chm.

The Aspect Workforce Web Services Help is displayed.

To use a test tool that enables developers to interact with the Aspect Workforce web services, Controller Tester User Guide.

About the Web Services Servers

In the online Help, web services are grouped by server name. The server name denotes the general purpose of the web services associated with it. Table 2 describes these servers.

Table 2: Servers for Web Services

Server	Use
Accounts	Account information, such as personal accounts and group allowance accounts.
Administration	Administrative functions, such as user profiles and authentication.
Agent Productivity	Information related to the Agent Productivity application of Perform.
Checker	Functionality related to the Checker system service, such as Checker rules and request evaluation.
Employees	Employee information, such as skills and employee groups.
Forecasting	Forecasting information, such as override sets.
IDP	Functionality related to the Intra-Day Performance (IDP) module, such as staff adjustments and intra-day performance forecasts.
Import Map	Site-mapping information used with Aspect Workforce™ Encompass.
Notification Server	Configuration information for the Notification Server application of Empower.
Pop Messages	Pop messaging system.
Schedule Preferences	Schedule preference information.
Official Schedules	Official schedule information, such as segment codes and superstates.
Seat Management	Seat management information, such as seat functions and seat groups.
Shift Bidding	Shift bidding information.
Employee Trades	Employee trade functionality, such as creating, processing, and canceling a trade request.
Trial Schedules	Trial schedule information.
Web Service Common	XML fragment definitions used in common by several web services.

Using the Web Services for Specific Tasks

To use the web services to accomplish specific tasks, see the Web Services online Help, included with this SDK. The Help provides overview topics (called functions lists) that enable you to see at a glance the specific task that each web service performs.

In the Help, the web services are grouped by the server with which they are associated. The Accounts Server, for example, has thirteen associated web services.

To use the web services for specific tasks:

1. Use Table 2 to decide which server the task is likely to be classified with.
2. Launch the **online Help.**

For instructions, see Launching the Online Help.

3. Click the **Contents** tab, and find the **server name** in the Table of Contents.
4. Click the **plus sign** next to the server name. A list of Help topics is displayed.
5. Click the **first topic** in the list, which is a functions list.
6. View the **list** of functions and descriptions associated with the server.
7. Click the **link** of the desired function to view detailed documentation about the function.

How to Make a Web Service Request

This section provides a sample script for making a web service request and instructions for customizing the script for your deployment.

Installing the Web Services

Before making a web service request, make sure you have installed the web services using the **WFMWebServices installer**.

The Alvaria Web Services are included with some Aspect Workforce enhancement packages, such as Empower and Encompass.

Sample Script

Following is a sample web service script. Start with this script, and customize it for your deployment, as described in Customizing the Sample Script.

```
/*
  Set m_WFMHost, m_WFMDatabaseAlias, m_Username, and m_Password
  according to your local configuration.
  Your installation of the WFM web services MUST (!) be set to use
BASIC authentication rather than FORM authentication.
*/
var m_WFMHost = 'reddoor';
var m_WFMDatabaseAlias = 'MSSQL_EWFM72';
```

```
var m_Username = 'TESTEMP1';

var m_Password = 'qqq';

var m_TimeoutValue = 480; // 8 minutes

function PostURL(ServletName)

{

  var result;

  result = 'http://' + m_WFMHost + '/WFMWebSvcs/' +
m_WFMDatabaseAlias + '/ENU/API/servlet/' + ServletName + '.ewfm';

  return result;

}


function PostBody(DataIn)

{

  var result;

  result = 'data_in=' + escape(DataIn) + '&Timeout=' +
m_TimeoutValue;

  return result;

}


function CallEWFMWebService(HttpReq, ServletName, DataIn)

{

  var HTTPREQUEST_SETCREDENTIALS_FOR_SERVER = 0;

  HttpReq.SetTimeouts(0, 0, 0, 0);

  // using POST method

  var url = PostURL(ServletName);

  var body = PostBody(DataIn);

  HttpReq.Open('POST', url);

  HttpReq.SetRequestHeader("Content-Type", "application/x-www-form-
urlencoded");

  HttpReq.SetCredentials(m_Username, m_Password,
HTTPREQUEST_SETCREDENTIALS_FOR_SERVER);

  HttpReq.Send(body);

  return HttpReq.ResponseText;

}


try

{

  var HttpReq = new ActiveXObject('WinHttp.WinHttpRequest.5.1');

  var x;


  x = CallEWFMWebService('AdminGetUserProfile', '');
```

```
  WScript.Echo(x);


/*
  x = CallEWFMWebService(HttpReq, 'AdminAgentDataGroupLookup',
'<AgentDataGroupLookup><PartialCode>A</PartialCode></AgentDataGroupLo
okup>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminContactDataGroupLookup',
'<ContactDataGroupLookup><PartialCode>A</PartialCode></ContactDataGro
upLookup>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'EmployeeInfo',
'<EmployeeInfo><Employee SK="-989928946686"></Employee><Employee
SK="-
989999973533"></Employee><IncludeExtraFields></IncludeExtraFields></E
mployeeInfo>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminGetUserProfile', '');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminEcho', '<Testing/>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminVersionInformation', '');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminSysInfo', '');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminSystemParameters',
'<SystemParameters><SystemParameter
Name="ForecastUnit"/></SystemParameters>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminSecurityLocks',
'<SecurityLocks><SystemLock><Lock>149</Lock></SystemLock><SystemLock>
<Lock>150</Lock></SystemLock></SecurityLocks>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminTimeConversion',
'<TimeConversion><TimeZone SK="-989999999596"/><DateTime>2001-
102T07:00:00</DateTime></TimeConversion>');
```

```
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AdminDatabaseAliases', ''));
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'EmployeeLookup',
'<EmployeeSelector><Employee SK="-
992147439297"/></EmployeeSelector>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'EmployeeGroups',
'<EmployeeGroupLookup><PartialCode>CSV</PartialCode></EmployeeGroupLo
okup>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'AccountPersonalAccountLookup',
'<PersonalAccountLookup><PartialCode>V</PartialCode></PersonalAccount
Lookup>');
  WScript.Echo(x);


  x = CallEWFMWebService(HttpReq, 'EmployeeLookup',
'<EmployeeSelector><Name>Zu</Name></EmployeeSelector>');
  WScript.Echo(x);


*/


}
catch(er)
{
  WScript.Echo('An error occurred: ' + er.description);
}
```

Customizing the Sample Script

Customize the sample script by setting the parameters according to your local configuration. The parameters are described in the following table.

Parameter	Description
m_WFMHost	Web server where the Alvaria Web Services are installed.
m_WFMDatabaseAlias	Aspect Workforce database alias.

Parameter	Description
m_Username	Username used to authenticate to the Alvaria Web Services. The user can be either a local account on the web server or a domain account. The user also must be entered in the Users module in Aspect Workforce.
m_Password	Password of user.

Recommended XML Coding Practices

Following are XML coding practices that Alvaria recommends for use with the SDK. Using these practices also helps ensure that your code is compatible with future versions of the SDK.

- Use a standard XML parsing technology, such as XML DOM or SAX.

Manual parsing of the XML returned by the Aspect Workforce web services can be a tedious and error-prone process.

- Take extreme care when dealing with strings of unknown content.

Constructing XML by string concatenation is less tedious than parsing, but it, too, can be prone to errors. For example, strings that might contain reserved XML characters must be encoded or enclosed in a CDATA section. A technology such as the XML DOM can alleviate many problems related to parsing.

- If possible, do not put a critical dependency on the length of string data types from an XML document.

For example, the XML schema might state that a string data type will be a maximum of x characters long. In a future version of the SDK, it could be necessary for Alvaria to increase the length of that string.

- Do not depend on the position of attributes in an XML node. Instead, identify attributes by their tag name.

For example:

```
<ANode Attribute1=”abc” Attribute2=”xyz”/>
```

is functionally equivalent to:

```
<ANode Attribute2=”xyz” Attribute1=”abc”/>
```

- Understand that XML, by definition, is extensible.

In future releases of the SDK, Alvaria might extend the content of the XML responses to accommodate new Aspect Workforce features or functionality. For example, in the current SDK release, a web service response might be documented as follows:

```
<ANode>
<ASubNode1>abc</ASubNode1>
</ANode>
```

But in a later release, the response might be extended to the following:

```
<ANode>
<ASubNode1>abc</ASubNode1>
<ASubNode2>xyz</ASubNode2>
</ANode>
```

- Ideally, your application should ignore any extensions that it does not understand or expect.

Wherever possible:

- Do not identify a node based on its position within the parent; instead, identify all nodes by their tag name.
- Do not depend on a node's having a fixed number of child nodes (as shown in the preceding example).
- Understand the ISO 8601 date and time representation standard, and develop your application so that it can read dates and times in any of the allowed formats.
- Be aware that the Aspect Workforce web services always expect that numeric values are formatted with a period as the decimal character, and that the web services always return numeric values, in the response, in that same format.

This is important if your application might run on a system where a character other than a period is the decimal character.

- Be aware that, internally, the Aspect Workforce application always references entities using their assigned surrogate key (SK) values.

For example, within the Aspect Workforce application, an employee is identified by an SK, whereas a human user typically identifies an employee by his ID. Your application might be affected if it references the value of an ID and then the Aspect Workforce user changes that ID. But the SK value never changes for the life of an entity.

This rule also applies to code values. For example, a staff group might have the code value CSVORD, representing customer service at O'Hare Airport. If the staff group moves to Midway Airport, the human user might change the code value to CSVMDW, so that the new code value is more appropriate to the new location. But internally, the SK value of this staff group does not change even when the code value changes. So if you reference the SK value rather than the code value, it is less likely that your application will be negatively affected.

- Every call to an Alvaria web service requires that the .ewfm extension be appended to the web service name. For more information, see How to Make a Web Service Request.

Controller Tester User Guide

This chapter describes the Aspect Workforce Controller Tester (the Controller Tester). The Controller Tester, included with the SDK, is a utility that enables developers to interact with the Alvaria Web Services. This guide provides detailed descriptions of the user interface of the Controller Tester and guidelines for the effective use of the tool.

Prerequisites

Table 1 shows the requirements for using the Controller Tester. These requirements are identical to those for using the Alvaria Web Services themselves.

Table 1: Aspect Workforce Requirements

Versions	Aspect Workforce 7.5 or later
Enhancement Packages	None additional

Audience

This guide is intended for developers who want to learn about and use the Alvaria Web Services.

Users of this guide should be familiar with the following:

- The HTTP protocol; especially how to make an HTTP POST request
- How to program with basic scripting languages, such as JavaScript or VB Script
- XML schemas; especially how to create an XML request using an XML schema, and how to process the XML response to extract the requested information using the Document Object Model (DOM) or Simple API for XML (SAX)

For detailed documentation of the Alvaria Web Services, see Chapter 6, Using the Web Services.

Overview

The Aspect Workforce Controller Tester is a Windows application that provides an easy way to interact with the Alvaria Web Services. The Controller Tester was originally developed at Alvaria as an internal tool for Aspect Workforce developers.

The application has options for saving and loading the web service parameters to and from a file. By default, the files have a .WFC extension. Although the .WFC files are text files, they should not be edited by hand, because hand-editing could damage the structure of the files and make them unusable by the application.

Launching the Utility

Launch the Controller Tester from the installation folder.

To launch the Controller Tester:

1. Navigate to the folder where the utility is installed. The default path is:
2. C:\Program Files (x86)\
   Alvaria\Workforce 25.0 SDK\Web_Services_Related_Files
3. Open the folder, and double-click the file WFMControllerTester.exe. The utility opens.
4. To view an online Help system that documents all web services in the SDK, see Launching the Online Help

About the Main Window

The main window of the Controller Tester consists of the following components:

- Service Selection Panel
- Tabs
- Menu Bar and Toolbar
- Status Bar

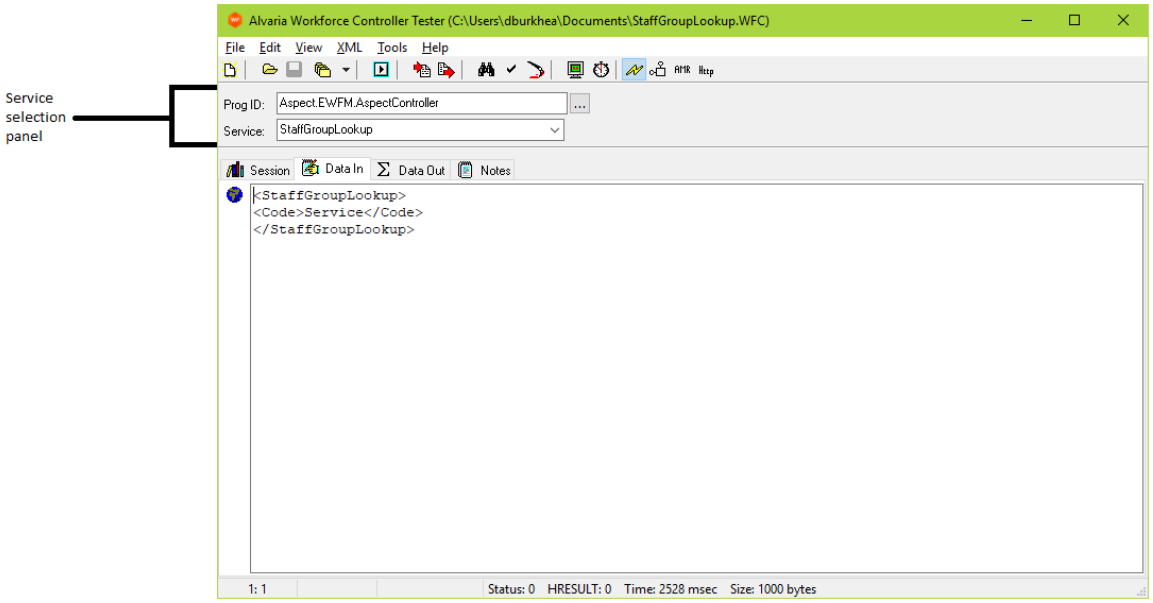


Figure 1: Aspect Workforce Controller Tester Main Window

Service Selection Panel

The service selection panel consists of the ProgID and Service fields. This panel designates which service you want to execute. It contains an edit box (with a lookup button) to select the Prog ID and a combo box to select the Service.

- ProgID field: If the Aspect Workforce controller has been installed and registered on your workstation, the lookup button can be used to select it from a list. Otherwise, you must enter the Prog ID manually. The Prog ID is the human-understandable form of the ClassID of an Aspect Workforce Controller COM object. The only Prog ID for this SDK is Aspect.EWFM.AspectController

The Aspect.EWFM.AspectController.dll is installed automatically on the computer when you install the main or secondary application server using the Aspect Workforce installation program.

- Service field: The Service combo box is used to select which service (as provided by the selected Controller Prog ID) you want to invoke.

Tabs

The Controller Tester has four tabs:

- Session Tab
- Data In Tab
- Data Out Tab
- Notes Tab

The first three tabs are intended to contain XML and provide both a raw and a rendered view of the XML data. The button to the left of the display (marked by a globe graphic) is used to toggle between raw and rendered XML mode. The XML can be edited only when the display is set to raw mode. The last tab, Notes, is for free-form text only.

Session Tab

The Session tab is used to enter the session context XML for the service. The session context XML contains information about the database alias to connect to and user information.

The contents of the Session tab are ignored when you call a service using the HTTP transport method.

Data In Tab

The Data In tab is used to enter the input XML for the web service.

Data Out Tab

The Data Out tab displays the output XML from the web service. The Data Out XML can be edited, but editing it is rarely useful.

Notes Tab

The Notes tab provides a place to enter any type of text. This text can be saved along with the other parameters. The Notes tab can be thought of as a convenient scratch pad for copying and pasting blocks of XML.

An example of a note might be:

“Retrieve schedule segments for Employee ID 2222 for Dec 4 thru 11, 2008”

Menu Bar and Toolbar

This section describes the menu bar and toolbar of the Controller Tester. All of the application's functions can be invoked from the menu bar, and many common menu functions can also be performed by using the toolbar buttons. A fly-over hint for each toolbar buttons displays the corresponding menu option, indicating the function that each button performs.

Viewed from left to right across the toolbar, the toolbar buttons and their menu bar equivalents are shown in Table 7-2**.** Menu options without toolbar equivalents are also shown in their logical place.

Table 2: Toolbar Buttons and Corresponding Menu Functions

Toolbar Button	Menu Function
	File > New
	File > Open
	File > Save
None.	File > Save As
	File > Reopen
None.	File > Revert
	File > Execute
	XML > Load From File
	XML > Save To File
None.	Edit > Cut/Copy/Paste/Delete/Select All
	Edit > Find
None.	Edit > Find Again
	View > Stay On Top
	Tools > Performance
	Tools > Execute Via Direct COM
	Tools > Execute Via COM Transport
	Tools > Execute Via Alvaria Message Routing (AMR) Transport
	Tools > Execute Via HTTP
None.	Tools > Message Transport Options
None.	Tools > HTTP

Toolbar Button	Menu Function
None.	Tools > Edit Servlet Map
None.	Tools > Ping
None.	Tools > Release Controller Interface
	Tools > Validate With Schema...
None.	Help > About

File Menu Options

This section describes the options available on the File menu.

File > New

This option clears all web service input parameters (that is, ProgID, Service, Session, and Data In).

File > Open

This option allows you to load a previously saved parameter file. The standard Windows File > Open dialog box is displayed

File > Save

This option saves the parameters to the currently loaded .WFC file. It is enabled only if a .WFC file has been previously opened and there are unsaved changes.

File > Save As

This option saves the parameters to a new or different .WFC file, whichever you prefer.

File > Reopen

This option displays, in a sub-menu, a list of the recently used .WFC files. The list of recently used files is maintained in the Windows registry.

File > Revert

This option undoes any edits made since the .WFC file was last saved. It is enabled only if there are unsaved changes.

File > Execute

This option invokes the web service call.

XML Menu Options

This section describes the options available on the XML menu.

XML > Load From File

This option allows you to load XML into the selected tab from a file. Unlike the .WFC files, which contain all of the web service parameters, this file should contain only the XML contents for the current tab. Its operation is equivalent to opening the selected file in Notepad, copying the entire contents to the clipboard, and pasting the contents from the clipboard into the XML edit. This option is enabled for the Session and Data In tabs. It is not available for rendered XML.

XML > Save To File

This option allows you to save XML from the selected tab to a file. Unlike the .WFC files, which contain all of the web service parameters, this file will contain only the XML contents of the current tab. Its operation is equivalent to copying the XML to the clipboard, opening Notepad, and pasting the XML from the clipboard into Notepad. This option is enabled for the Session, Data In, and Data Out tabs. It is not available for rendered XML.

Edit Menu Options

This section describes the options available on the Edit menu.

Edit > Cut/Copy/Paste/Delete/Select All

These options provide the standard Windows clipboard operations.



Note: The menu options in this group operate on raw XML only. They do not operate on rendered XML. You can operate on rendered XML by using a web browser control that is part of Internet Explorer. This control includes a pop-up menu that provides a number of options for rendered XML, including clipboard operations.

Edit > Find

This option displays the standard Windows Find dialog box that allows you to search for specific text. This option is not available for searching rendered XML.

Edit > Find Again

This option allows you to repeat the last Find operation. It is enabled only after a Find operation has been performed.

View Menu Option

This section describes the option available on the View menu.

View > Stay On Top

This option toggles the position of the application's main window in relation to other applications that are currently running. When this option is selected, the Controller Tester's main window will be on top of all other windows.

Tools Menu Options

This section describes the options available on the Tools menu.

Tools > Performance

This option displays a dialog box (shown in Figure 7-2) that allows you to test the performance of a selected service. It is not available if you use HTTP invocation.

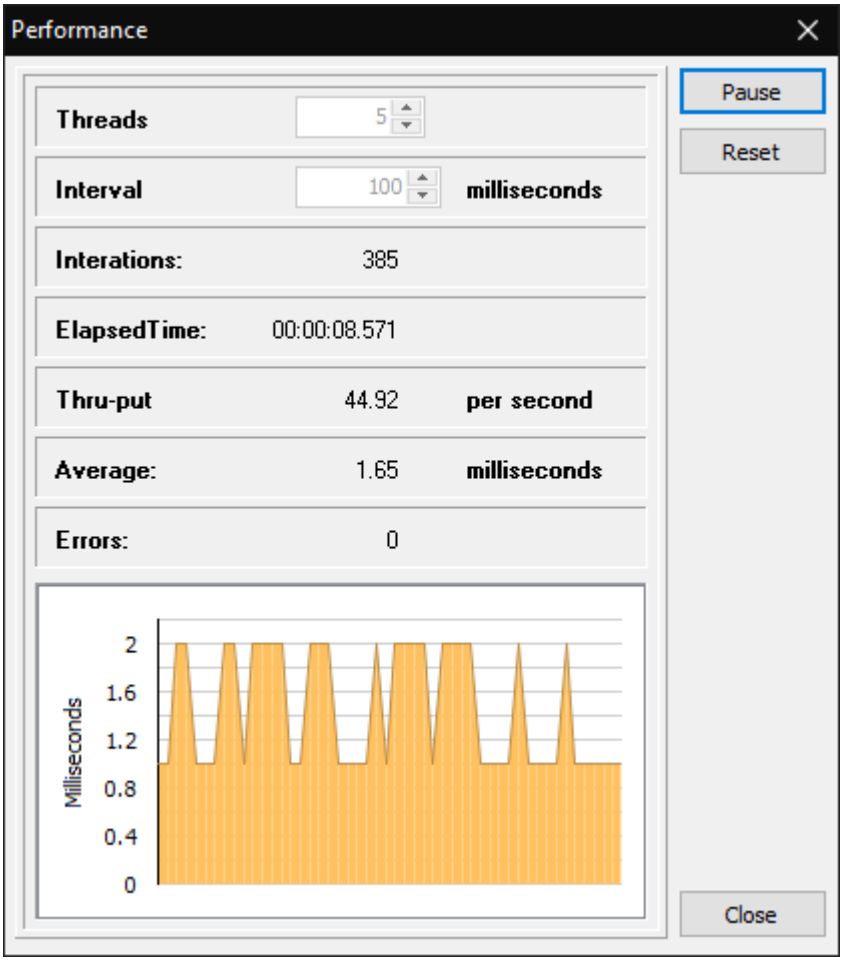


Figure 2: Tools > Performance Dialog Box

The graph plots the service call times in a scrolling window, conveying a rough depiction of how the service is performing. But at high thru-put levels, the graph may not be able to keep pace with the performance of the service.

Table 3 shows the terms in the dialog box with a description of each.

Table 3: Performance Dialog Box Terms

Term	Description
Threads	Specifies how many threads are created to call the service.
Interval	Specifies the delay, if any, between service calls on each thread.
Start/Pause/Resume	Click the Start button to start the performance test. The button label changes to *Pause* when the test has started, and changes to *Resume* when the test has been paused.
Reset	Click this button to reset the performance metrics.
Iterations	Total number of service calls that have been made across all threads.

Term	Description
Elapsed Time	Total elapsed clock time of the test.
Thru-put	Average number of service calls per second.
Average	Average time (in milliseconds) that each service call took. The value of (Average * Iterations) can be greater than the value of Elapsed Time because of multiple threads.
Errors	A count of the number of error responses, including timeouts.
Close	Click this button to close the dialog box.

Tools > Execute Via HTTP

This option invokes the web service using the HTTP transport method. It is the usual means for accessing **the web service.**

Tools > Execute Via Direct COM / COM Transport / AMR Transport

These options are primarily for internal use by Alvaria. To use any of the options, the Aspect Workforce Controller, Transport, and Alvaria Message Routing libraries must be installed on your workstation.

Tools > Message Transport Options

This option allows you to configure the transport options. It should be used only when either the COM Transport or AMR Transport option is selected. The transport configuration options are maintained in the Windows registry.

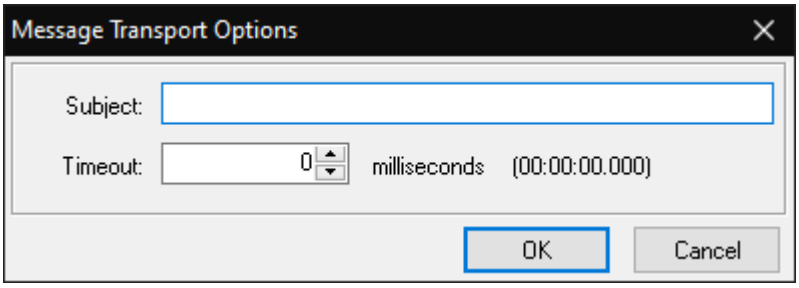


Figure 3: Tools > Message Transport Options Dialog Box

Tools > HTTP

This option displays a dialog box for configuring how the service is invoked using HTTP. The contents of the Session tab are ignored when you call a web service using the HTTP transport method. For more information about using this transport method, see Choosing Servlet Names with HTTP.

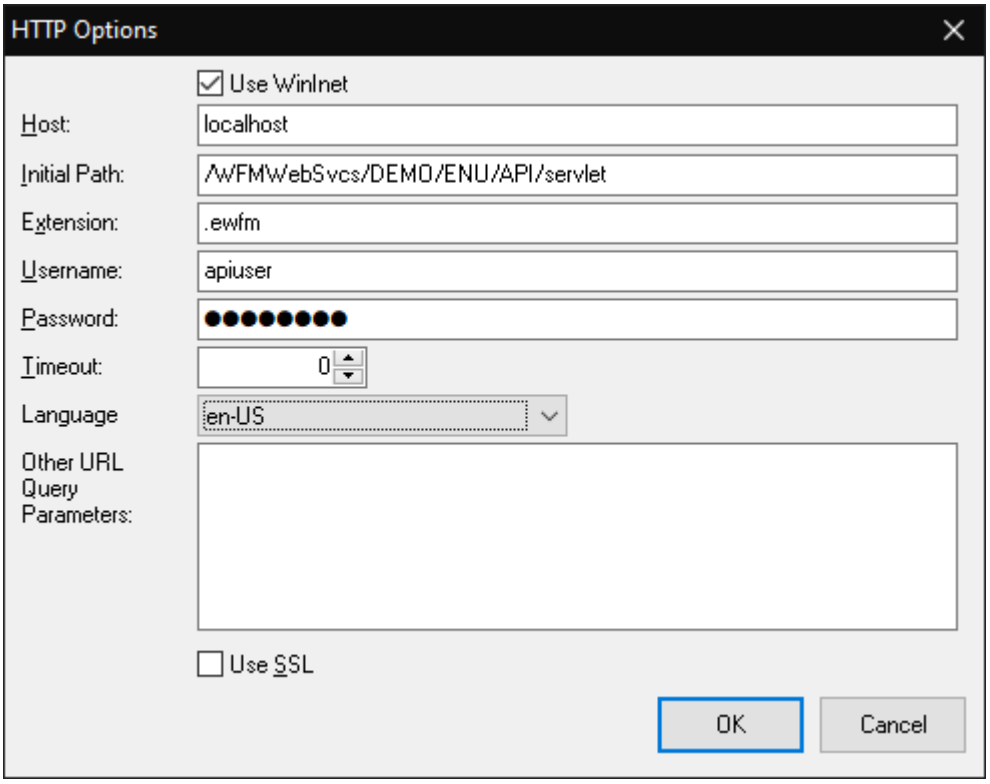


Figure 4: Tools > HTTP Dialog Box

Table 4 shows the configuration options with guidelines for using each.

Table 4: HTTP Configuration Options

Option	Guidelines
Use WinInet	Check the check box if you want the WinInet interface to handle authentication. This box should always be checked.
Host	Type the machine name of your HTTP server.
Initial Path	Type the path on the HTTP Server up to *but not including* the servlet name. The path should be in the following format, where ***<DBALIAS>*** is a valid Aspect Workforce database alias name: `/WFMWebSvcs/<DBALIAS>/ENU /API/servlet`
Extension	Leave this set to the default value: **.ewfm**
Username	Username used to authenticate to the Alvaria Web Services. The user can be either a local account on the web server or a domain account. The user also must be entered in the Users module in Aspect Workforce.
Password	Password of user.

Option	Guidelines
Timeout	Type the timeout (in seconds) within which the servlet should return the response.
Language	Select a language to include in the HTTP request. The language is specified with a Language=<lang> URL query parameter. If the response of a web service includes localized strings, those strings will be presented in the specified language. If no language is specified, the language of the operating system where the WFM Dispatcher service is installed will be used.
Other URL Query Parameters	You can optionally specify additional URL query parameters to include in the HTTP request. These additional parameters will be appended to any other URL query parameters for the request, so you should prepend an ampersand (&) before the first URL query parameter. For example: &param1=value1&param2=value2
Use SSL	Select the check box if you want to use HTTPS instead of HTTP when calling the web service.

These configuration options are maintained in the Windows registry.

Tools > Edit Servlet Map

This option allows you to modify the servlet name that is mapped to a Prog ID/Service pair. These configuration options are maintained in the Windows registry. Figure 5 shows the dialog box invoked by this option.

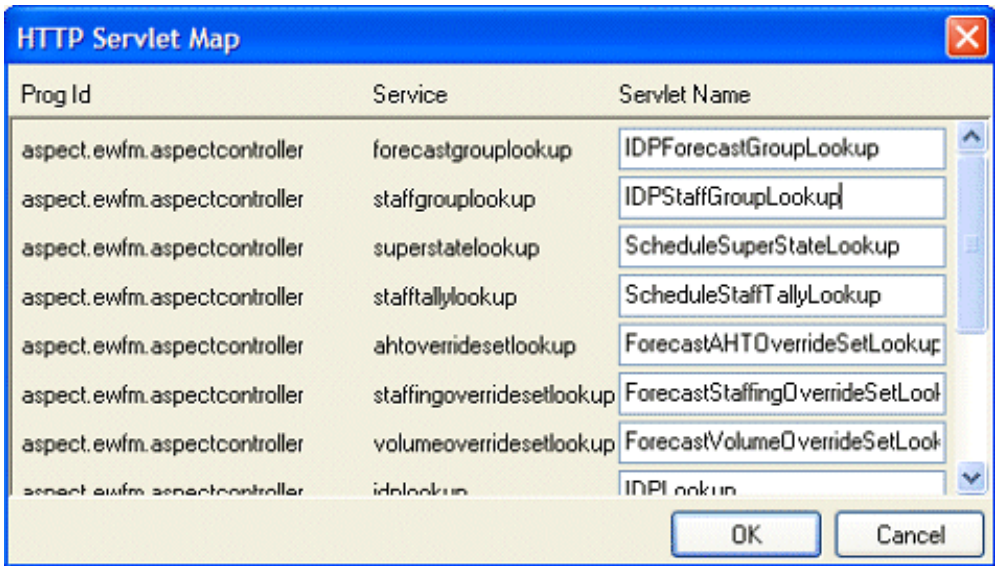


Figure 5: Tools > Edit Servlet Map Dialog Box

Tools Release Controller Interface

This option frees the instance of the controller COM object.

Tools > Ping

This option is primarily for internal use by Alvaria. It shows the machine name, process ID, and thread ID of the servers responding to the selected service call.

Tools > Validate With Schema

This option validates the given XML against an XML schema.

XML is said to be *invalid* when it violates one or more rules that are defined in the schema. Points to remember:

- Sometimes a request containing invalid XML returns a valid response.
- When you try to validate this invalid XML using the Validate With Schema option, the Controller Tester returns an error that specifies why the XML is invalid.

Help Menu Option

This section describes the option available on the Help menu.

Help > About

This option shows the current version of the application.

Status Bar

The status bar at the bottom of the application window display status information about the application, such as:

- Row/column position (appears only during editing of raw XML)
- Indication of whether the current .WFC file has been modified
- Current activity of the application (Executing, Rendering XML, and so on)
- Information about the most recent execution (including HRESULT, time of execution, and size of the response)

Choosing Servlet Names with HTTP

The first time you try to call a web service using the HTTP transport method, the Controller Tester prompts you to enter the servlet name. The servlet name is the full name of the web service. This information is necessary because the full name of the web service is usually not the same as the service name that is displayed in the service selection panel. (For information about this panel, see Service Selection Panel.)

Typically, the web service name consists of:

- A prefix consisting of the name or acronym of the server with which the web service is associated. (For a list of these servers, see About the Web Services Servers.)
- The service name as displayed in the service selection panel.

For example, consider the format of the servlet name shown in Figure 6:

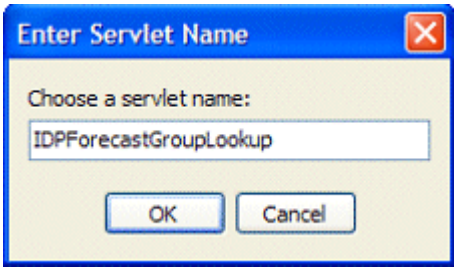


Figure 6: Typical Format of Servlet Name

The service name of this web service is **ForecastGroupLookup**; it is associated with the server named **IDP**. So the servlet name you should enter for this web service is **IDPForecastGroupLookup**.

This naming convention holds true in the vast majority of cases, but not always. Two exceptions are the servlets named IDP and IDPLookup. These two servlet names are identical to their service names.

The most reliable way to determine the servlet name is to find the name in the Web Services Help. For each server, the servlet names are listed in the functions list. For instructions, see Using the Web Services for Specific Tasks.

Intra-Day Performance Column Descriptions

This chapter describes the columns that are displayed in the Intra-Day Performance (IDP) module of Aspect Workforce.

Audience

The chapter is intended to be used by developers when working with the IDP function, which is documented in the web services Help system in this SDK.

Overview

Intra-day performance (IDP) columns are categories of data that are available for display in an intra-day performance forecast in Aspect Workforce. Descriptions about these IDP columns are given here in the form of statistics, detailed descriptions, and calculations. In Aspect Workforce, IDP forecasts are accessed through the Intra-Day Performance and View Intra-Day Performance modules.

For developers, IDP columns relate to the IDP web service. The IDP web service is one of the Alvaria Web Services, grouped under the IDP Server. The information provided for each IDP column—in particular the Internal Field Name—is needed to request the correct data when making XML requests with the IDP web service.

For more information about the IDP web service, see About the IDP Web Service and the Aspect Workforce Web Services online Help (included with this SDK). To access the Help, see Launching the Online Help.

About the IDP Web Service

When you use the IDP web service, the request is an XML document that contains information defining what intra-day performance forecast information is requested. (For example, the Revised Forecast Volume column appears in the Aspect Workforce interface under the default title name RVOL. But the internal field name to be used in the request is RvsForVol.) The server parses the request document and retrieves detailed IDP information from the IDP forecast identified by the SK on the IDP node of the request.

Using the IDP Web Service with Encompass

The intra-day performance data in the XML output can be used with Encompass to import IDP data from a remote site into the local site. This can only occur when mapping information has associated the staff groups and forecast groups at one site with the corresponding groups at the other site.

To integrate a site that is running Encompass with a site that is not running Aspect Workforce, the XML structure must match a specific structure. The structure is documented in the Encompass Web Services online Help. In the Intra-Day Performance Server chapter of the Help, see the topic for the IDP function.

Information Provided

For each intra-day performance column, the following information is typically provided:

- Default title name of the column followed, on the next line, by the informal column name (such as RVOL / Revised Forecast Volume)
- Detailed description, including how the statistic is calculated (if applicable)
- Table with technical information about the column (such as Min. Value and Max. Value)

The information is divided into sections for forecast group statistics and staff group statistics.

Forecast Group Statistics

This section provides descriptions of intra-day performance columns that pertain primarily to forecast group statistics.

AABD

Actual Calculated Abandoned

The actual number of contacts received by the ACD but disconnected by the customer before an agent answers. This value is calculated as Actual Number of Contacts Offered minus Actual Number of Contacts Handled. The Summary value is a total.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	AABD
Internal Field Name	ActualAbandoned
Description	Actual Abandoned
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AAHT

Actual AHT

The average handle time (talk time + wrap-up time) reported by the ACD. If you edit this value, the AAHT field for that period becomes locked against a refresh. The Summary value is an average weighted by Actual Number of Contacts Handled.

This value is highlighted if it falls outside of the designated AHT threshold, which for this statistic compares the Revised Forecast AHT to the Actual AHT.

In consolidated mode, this value is an average weighted by each forecast group's Actual Number of Contacts Handled.

When using a contact allocation configuration with Aspect Workforce™ Allocate, this column is blank for parent forecast groups. AHT is available only for child forecast groups that are routed to a staff group.

Default Title Name	AAHT
Internal Field Name	ActualAHT
Description	Actual AHT
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	9,999,999.99
Table Name	ITD_FOR_CALL_STAT
Field Name	AHT

ABCKLOG

Actual Calculated Backlog

The estimated number of contacts remaining in queue at the end of the period, calculated based on actual data reported by the ACD. The Summary value is a simple average per period, as is the partial-day summary value. In consolidated mode, the value for each period is a total for the consolidated groups, and the simple averages are calculated from these consolidated values.

This statistic is available only for forecast groups in a multiskill routing set or forecast groups configured for long service delays. It is calculated only when you use the Calculate Statistics option.

In an Allocate contact allocation configuration, a parent group's value is the sum of the calculated values for its child groups.

ABCKLOG J

Actual Calculated Backlog with Staff Adjustments

The estimated number of contacts remaining in queue at the end of the period, calculated based on actual data reported by the ACD and with staff adjustments taken into account. The Summary value is a simple average per period, as is the partial-day summary value. In consolidated mode, the value for each period is a total for the consolidated groups, and the simple averages are calculated from these consolidated values.

This statistic is available only for forecast groups in a multiskill routing set or forecast groups configured for long service delays. It is calculated only when you use the Calculate Statistics option.

In an Allocate contact allocation configuration, a parent group's value is the sum of the calculated values for its child groups.

ABD-ACD

Actual Abandoned – From ACD

The actual number of connections that were disconnected by the customer before an agent answered, as reported by the ACD. The Summary value is a total.

Periods can be edited. If a period is edited, it becomes locked.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	ABD-ACD
Internal Field Name	ActualABD
Description	Actual Abandons
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	ITD_FOR_CALL_STAT
Field Name	ABD

Actual Simulated Queue Backlog With Adjustments

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	TSimForActBacklog1
Description	Forecast group’s Simulated Actual Queue Backlog With Adjustments
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	9,999,999
Table Name	NA
Field Name	NA

Actual Simulated Queue Backlog Without Adjustments

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	TSimForActBacklog2
Description	Forecast group's Simulated Actual Queue Backlog Without Adjustments
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	9,999,999
Table Name	NA
Field Name	NA

ADELAY HR

Actual Calculated Average Delay 2: Hours

The calculated average delay between the time a contact is received by the ACD and the time it is answered by an agent based on actual contacts, in hours. This column is calculated based on: Actual Number of Contacts Offered, Actual AHT, Tendency to Abandon, and Forecast Group's Actual Scheduled Without Staff Adjustments And Unproductive. The Summary value is an average weighted by the Actual Number of Contacts Handled.

In consolidated mode, this value is an average weighted by each forecast group's ANCH.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ADELAY HR
Internal Field Name	ActualCalcAvgDelay2Hour
Description	Actual Calculated Average Delay 2 Hour (based on Actual Calls Offered, Actual AHT, and the forecast group's Simulated Actual Scheduled Without Adjustments Without Unproductive) The Actual Calculated Average Delay 2 converted to hours.
Min. Decimals	0
Max. Decimals	1
Min. Value	0

Max. Value	48.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ADELAY J HR

Actual Calculated Average Delay 1 (with Staff Adjustments): Hours

The calculated average delay between the time a contact is received by the ACD and the time it is answered by an agent based on actual contacts, in hours. This column is calculated based on: Actual Number of Contacts Offered, Actual AHT, Tendency to Abandon, and Forecast Group's Actual Scheduled With Staff Adjustments And Without Unproductive. The Summary value is an average weighted by the Actual Number of Contacts Handled.

In consolidated mode, this values is an average weighted by each forecast group's ANCH.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ADELAY J HR
Internal Field Name	ActualCalcAvgDelay1Hour
Description	Actual Calculated Average Delay 1 Hour (based on Actual Calls Offered, Actual AHT and the forecast group's Simulated Actual Scheduled With Adjustments Without Unproductive). The Actual Calculated Average Delay 1 converted to hours.
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	48.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ADELAY J MIN

Actual Calculated Average Delay 1 (with Staff Adjustments): Minutes

The calculated average delay between the time a contact is received by the ACD and the time it is answered by an agent based on actual contacts, in minutes. This column is calculated based on: Actual

Number of Contacts Offered, Actual AHT, Tendency to Abandon, and Forecast Group's Actual Scheduled With Staff Adjustments And Without Unproductive. The Summary value is an average weighted by the Actual Number of Contacts Handled.

In consolidated mode, this value is an average weighted by each forecast group's ANCH.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ADELAY J MIN
Internal Field Name	ActualCalcAvgDelay1Minute
Description	Actual Calculated Average Delay 1 Minute (based on Actual Calls Offered, Actual AHT and the forecast group's Simulated Actual Scheduled With Adjustments Without Unproductive). The Actual Calculated Average Delay 1 converted to minutes.
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	2880.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ADELAY J SEC

Actual Calculated Average Delay 1 (with Staff Adjustments): Seconds

The calculated average delay between the time a contact is received by the ACD and the time it is answered by an agent based on actual contacts, in seconds. This column is calculated based on: Actual Number of Contacts Offered, Actual AHT, Tendency to Abandon, and Forecast Group's Actual Scheduled With Staff Adjustments And Without Unproductive. The Summary value is an average weighted by the Actual Number of Contacts Handled.

In consolidated mode, this values is an average weighted by each forecast group's ANCH.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ADELAY J SEC
Internal Field Name	ActualCalcAvgDelay1

Description	Actual Calculated Average Delay 1 (based on Actual Calls Offered, Actual AHT and the forecast group's Simulated Actual Scheduled With Adjustments Without unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	172800.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ADELAY MIN

Actual Calculated Average Delay 2: Minutes

The calculated average delay between the time a contact is received by the ACD and the time it is answered by an agent based on actual contacts, in minutes. This column is calculated based on: Actual Number of Contacts Offered, Actual AHT, Tendency to Abandon, and Forecast Group's Actual Scheduled Without Staff Adjustments And Unproductive. The Summary value is an average weighted by the Actual Number of Contacts Handled.

In consolidated mode, this values is an average weighted by each forecast group's ANCH.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ADELAY MIN
Internal Field Name	ActualCalcAvgDelay2Minute
Description	Actual Calculated Average Delay 2 Minute (based on Actual Calls Offered, Actual AHT, and the forecast group's Simulated Actual Scheduled Without Adjustments Without Unproductive) The Actual Calculated Average Delay 2 converted to minutes.
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	2880.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ADELAY SEC

Actual Calculated Average Delay 2: Seconds

The calculated average delay between the time a contact is received by the ACD and the time it is answered by an agent based on actual contacts, in seconds. This column is calculated based on: Actual Number of Contacts Offered, Actual AHT, Tendency to Abandon, and Forecast Group's Actual Scheduled Without Staff Adjustments And Unproductive. The Summary value is an average weighted by the Actual Number of Contacts Handled.

In consolidated mode, this value is an average weighted by each forecast group's ANCH.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ADELAY SEC
Internal Field Name	ActualCalcAvgDelay2
Description	Actual Calculated Average Delay 2 (based on Actual Calls Offered, Actual AHT, and the forecast group's Simulated Actual Scheduled Without Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	172800.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ANCA

Actual Outbound Attempts

The actual number of outbound attempts reported by the dialer. Results of an attempt include: right-party contact, wrong-party contact, abandoned by dialed party, line is busy, answering machine answered. The Summary value is a total.

Periods can be edited. If a period is edited, it becomes locked.

In consolidated mode, this value is sum of the individual values for each period.

Default Title Name	ANCA
Internal Field Name	ActualNCA
Description	Actual Contacts Attempted
Min. Decimals	0

Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	ITD_FOR_CALL_STAT
Field Name	NCA

ANCH

Actual Number of Contacts Handled

The actual number of contacts handled by an agent, as reported by the ACD. The Summary value is a total.

If you edit this value, it becomes locked. If this value is blank or zero and you edit the Actual Number of Contacts Offered, the ANCO value is copied to ANCH.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	ANCH
Internal Field Name	ActualNCH
Description	Actual Contacts Handled
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	ITD_FOR_CALL_STAT
Field Name	NCH

ANCO

Actual Number of Contacts Offered

The actual number of offered contacts reported by the ACD. The Summary value is a total.

If a period is edited, the ANCO field becomes locked.

This value is highlighted if it falls outside of the designated Volume threshold, which for this statistic compares Revised Forecast Volume to Actual Number of Contacts Offered.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	ANCO
Internal Field Name	ActualNCO

Description	Actual Calls Offered
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	ITD_FOR_CALL_STAT
Field Name	NCO

ANNC

Actual Non-Connections

The number of attempts that do not result in a connection. This includes attempts when an answering machine answered or the line was busy. (Right-party contacts, wrong-party contacts, and connections abandoned by the dialed party are all considered connections and are not counted for this statistic.) The Summary value is a total.

This column is calculated as follows:

Actual Outbound Attempts – Actual Contacts Offered

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	ANNC
Internal Field Name	ActualNNC
Description	Actual Non-connects
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AOABDPCT

Actual Outbound Abandoned Percent

The number of attempts abandoned by the dialed party compared to the total number of connections. (Connections can be right-party, wrong-party, or abandoned by the dialed party.) The Summary value is an average weighted by the Actual Contacts Offered in the individual periods.

This column is calculated as follows: (Actual Contacts Offered – Actual Total Contacts) / Actual Contacts Offered * 100

In consolidated mode, this value is an average weighted by each forecast group's Actual Contacts Offered.

Default Title Name	AOABDPCT
Internal Field Name	ActOABDPct
Description	Actual Outbound Calls Abandoned Percent *Calculation*: (Connected Calls - Total Contacts) / Connected Calls
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ARPC

Actual Right-Party Contacts

The actual number of right-party contacts, as reported by the dialer. The Summary value is a total.

Periods can be edited. If a period is edited, it becomes locked.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	ARPC
Internal Field Name	ActualRPC
Description	Actual Right-Party Contacts
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	ITD_FOR_CALL_STAT
Field Name	RPC

ARPCATTPCT

Right-Party Contacts to Attempts Percent

The percentage of right-party contacts compared to the total number of outbound attempts. The Summary value is an average weighted by the Actual Outbound Attempts.

This column is calculated as follows:

Actual Right-Party Contacts / Actual Outbound Attempts * 100

In consolidated mode, this value is an average weighted by each forecast group's Actual Outbound Attempts.

Default Title Name	ARPCATTPCT
Internal Field Name	ActRPCAttPct
Description	RPC/Attempt Percent *Calculation*: Right-Party Contacts / Contacts Attempted
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ARPCCONPCT

Right-Party Contacts to Total Connections Percent

The percentage of right-party contacts compared to the total number of connections. (Connections can result in right-party contacts, wrong-party contacts, or connections abandoned by the dialed party.) The Summary value is an average weighted by the Actual Contacts Offered.

This column is calculated as follows:

Actual Right-Party Contacts / Actual Contacts Offered * 100

In consolidated mode, this value is an average weighted by each forecast group’s Actual Contacts Offered.

Default Title Name	ARPCCONPCT
Internal Field Name	ActRPCConPct
Description	RPC/Connect Percent *Calculation*: Right-Party Contacts / Connected Calls
Min. Decimals	0

Max. Decimals	2
Min. Value	0.00
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ARPHT

Actual Right-Party Average Handle Time

The average handle time (talk time + wrap-up time) for Actual Right-Party Contacts, as reported by the dialer. The Summary value is an average weighted by Actual Right-Party Contacts.

Periods can be edited. If a period is edited, it becomes locked.

In consolidated mode, this value is an average weighted by each forecast group's Actual Right-Party Contacts.

For an Allocate configuration in which contacts are allocated, this column is blank for parent forecast groups. AHT is available only for child forecast groups that are routed to a staff group.

Default Title Name	ARPHT
Internal Field Name	ActualRPHT
Description	Actual Right-Party Average Handle Time
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	9,999,999.99
Table Name	ITD_FOR_CALL_STAT
Field Name	RPHT

ASL

Actual Calculated Service Level Percent 2

The calculated percentage of actual contacts answered within the acceptable delay (SLGOALSEC, SLGOALMIN, or SLGOALHR). This calculated service level is based on: Actual Contacts, Actual AHT, Tendency to Abandon, the Service Level Goal (in seconds, minutes, or hours), and Forecast Group's Actual Scheduled Without Staff Adjustments And Unproductive. The Summary value is an average weighted by Actual Number of Contacts Offered.

In consolidated mode, this value is an average weighted by ANCO.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ASL
Internal Field Name	ActualCalcSrvcLevPct2
Description	Actual Calculated Service Level Percent 2 (based on Actual Calls Offered [ANCO] and Actual AHT [AAHT] and the forecast group's Simulated Actual Scheduled Without Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ASL J

Actual Calculated Service Level Percent 1 (with Staff Adjustments)

The calculated percentage of actual contacts answered within the acceptable delay (SLGOALSEC, SLGOALMIN, or SLGOALHR). Staff adjustments are included. This calculated service level is based on: Actual Contacts, Actual AHT, Tendency to Abandon, the Service Level Goal (in seconds, minutes, or hours), and Forecast Group's Actual Scheduled With Staff Adjustments And Without Unproductive. The Summary value is an average weighted by the Actual Number of Contacts Offered.

In consolidated mode, this value is an average weighted by the ANCO.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ASL J
Internal Field Name	ActualCalcSrvcLevPct1
Description	Actual Calculated Service Level Percent 1 (based on Actual Calls Offered [ANCO] and Actual AHT [AAHT] and the forecast group's Simulated Actual Scheduled With Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	2

Min. Value	0
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ASLSEC

Actual Service Level Seconds

The acceptable delay, in seconds, before a contact is answered, as configured in the ACD. (This value is used to calculate the Actual Service Level percentage.) The Summary value is an average weighted by the Actual Contacts Offered in the individual periods.

Periods can be edited. If a period is edited, it becomes locked.

In consolidated mode, this value is an average weighted by each forecast group's Actual Contacts Offered.

Default Title Name	ASLSEC
Internal Field Name	ActualSLSec
Description	Actual Service Level Seconds
Min. Decimals	0
Max. Decimals	1
Min. Value	0.0
Max. Value	9999.9
Table Name	ITD_FOR_CALL_STAT
Field Name	SRVC_LEV_SEC

ATPC

Actual Total Contacts

The Actual Right-Party Contacts plus the Actual Wrong-Party Contacts. (Abandoned calls are not counted.) The Summary value is a total.

This column is calculated as follows:

Actual Right-Party Contacts + Actual Wrong-Party Contacts

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	ATPC
Internal Field Name	ActualTPC
Description	Actual Total Contacts

Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AWPC

Actual Wrong-Party Contacts

The Actual Contacts Handled minus the Actual Right-Party Contacts. The Summary value is a total.

This column is calculated as follows:

Actual Contacts Handled – Actual Right-Party Contacts

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	AWPC
Internal Field Name	ActualWPC
Description	Actual Wrong-Party Contacts
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AWPCATTPCT

Wrong-Party Contacts to Attempts Percent

The percentage of wrong-party contacts compared to the total number of attempts. The Summary value is an average weighted by the Actual Outbound Attempts.

This column is calculated as follows:

Actual Wrong-Party Contacts / Actual Outbound Attempts * 100

In consolidated mode, this value is an average weighted by each forecast group's Actual Outbound Attempts.

Default Title Name	AWPCATTPCT

Internal Field Name	ActWPCAttPct
Description	WPC/Attempt Percent *Calculation*: Wrong-Party Contacts / Contacts Attempted
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AWPCCONPCT

Wrong-Party Contacts to Connections Percent

The percentage of wrong-party contacts compared to the total number of connections. A connection can be a wrong-party contact, a right-party contact, or a connection abandoned by the dialed party. The Summary value is an average weighted by the Actual Contacts Offered.

This column is calculated as follows:

Actual Wrong-Party Contacts / Actual Contacts Offered * 100

In consolidated mode, this value is an average weighted by each forecast group's Actual Contacts Offered.

Default Title Name	AWPCCONPCT
Internal Field Name	ActWPCConPct
Description	WPC/Connect Percent *Calculation*: Wrong-Party Contacts / Connected Calls
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AWPHT

Actual Wrong-Party Average Handle Time

The average handle time (talk time + wrap-up time) for Actual Wrong-Party Contacts. The Summary value is an average weighted by Actual Wrong-Party Contacts.

This column is calculated as follows:

(Actual AHT * Actual Contacts Handled – Actual Right-Party AHT * Actual Right-Party Contacts) / Actual Wrong-Party Contacts

In consolidated mode, this value is an average weighted by each forecast group's Actual Wrong-Party Contacts.

For an Allocate configuration in which contacts are allocated, this column is blank for parent forecast groups. AHT is available only for child forecast groups that are routed to a staff group.

Default Title Name	AWPHT
Internal Field Name	ActualWPHT
Description	Actual Wrong-Party Average Handle Time
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	9,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

BACKLOG

Actual Email Backlog

The number of emails waiting to be answered, based on the email queue at the end of the period, as reported by the email system.

Periods can be edited. If a period is edited, it becomes locked.

Default Title Name	BACKLOG
Internal Field Name	ActualBacklog
Description	Actual Backlog
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999

Table Name	ITD_FOR_CALL_STAT
Field Name	BCKLOG

Balk Percentage

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	BalkPct
Description	Balk Percentage
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	100.00
Table Name	IDP_FOR_GRP_DET
Field Name	BALK_PCT

DELAY-ACDHR

Actual Average Delay: Hours

The average delay (number of hours customers are waiting for an agent) reported by the ACD. The Summary value is an average weighted by the Actual Number of Contacts Handled (ANCH).

In consolidated mode, this value is an average weighted by each forecast group's ANCH.

Default Title Name	DELAY-ACDHR
Internal Field Name	AvgDelayHour
Description	Actual Average Delay from the ACD.
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	48.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

DELAY-ACDMIN

Actual Average Delay: Minutes

The average delay (number of minutes customers are waiting for an agent) reported by the ACD. The Summary value is an average weighted by the Actual Number of Contacts Handled (ANCH).

In consolidated mode, this value is an average weighted by each forecast group's ANCH.

Default Title Name	DELAY-ACDMIN
Internal Field Name	AvgDelayMinute
Description	Actual Average Delay from the ACD.
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	2880.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

DELAY-ACDSEC

Actual Average Delay: Seconds

The average delay (number of seconds customers are waiting for an agent) reported by the ACD. The Summary value is an average weighted by the Actual Number of Contacts Handled (ANCH).

In consolidated mode, this value is an average weighted by each forecast group's ANCH.

Default Title Name	DELAY-ACDSEC
Internal Field Name	AvgDelaySec
Description	Actual Average Delay from the ACD.
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	172800.0
Table Name	ITD_FOR_CALL_STAT
Field Name	AVG_DELAY_SEC

DELAYGOALHR

Average Delay Goal: Hours

The desired number of hours between the time a contact is received by the ACD and the time it is answered by an agent. This value is specified in the forecasting scenario. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is calculated as follows: First, for any forecast group that has Service Level as its service quality goal, that goal is converted to an equivalent average delay to be consolidated with that of the other groups. Then all individual values are weighted by the Revised Forecast Volume and averaged.

This column is blank if the staffing basis is Workload or Service Level.

Default Title Name	DELAYGOALHR
Internal Field Name	AvgDelayGoalHour
Description	Average Delay Goal Hours
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	48
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

DELAYGOALMIN

Average Delay Goal: Minutes

The desired number of minutes between the time a contact is received by the ACD and the time it is answered by an agent. This value is specified in the forecasting scenario. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is calculated as follows: First, for any forecast group that has Service Level as its service quality goal, that goal is converted to an equivalent average delay to be consolidated with that of the other groups. Then all individual values are weighted by the Revised Forecast Volume and averaged.

This column is blank if the staffing basis is Workload or Service Level.

Default Title Name	DELAYGOALMIN
Internal Field Name	AvgDelayGoalMinute
Description	Average Delay Goal Minutes
Min. Decimals	0
Max. Decimals	1

Min. Value	0
Max. Value	2880
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

DELAYGOALSEC

Average Delay Goal: Seconds

The desired number of seconds between the time a contact is received by the ACD and the time it is answered by an agent. This value is specified in the forecasting scenario. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is calculated as follows: First, for any forecast group that has Service Level as its service quality goal, that goal is converted to an equivalent average delay to be consolidated with that of the other groups. Then all individual values are weighted by the Revised Forecast Volume and averaged.

This column is blank if the staffing basis is Workload or Service Level.

Default Title Name	DELAYGOALSEC
Internal Field Name	AvgDelayGoalSec
Description	Average Delay Goal Seconds
Min. Decimals	0
Max. Decimals	1
Min. Value	0.1
Max. Value	172800.0
Table Name	IDP_FOR_GRP_DET
Field Name	AVG_DELAY_GOAL_SEC

EXCL

Exclude from Reforecasting Indicator

This column contains a check box to indicate whether or not you have excluded the values in this period from being reforecast using the reforecast option from the Special menu of the Intra-Day Performance Detail window in an intra-day performance forecast.

Default Title Name	EXCL
Internal Field Name	ExclReforFlag
Description	Exclude this period for reforecasting indicator

Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	IDP_FOR_GRP_DET
Field Name	EXCL_REFOR_FLAG

FGANET

Forecast Group's Actual Net Staff 2

The number of scheduled staff over or under the actual number of staff required. This value is calculated as:

Forecast Group's Actual Scheduled Without Staff Adjustments And Unproductive – Forecast Group's Actual Required Without Unproductive

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistic compares the Forecast Group's Actual Scheduled Without Staff Adjustments to the Forecast Group's Actual Required Without Unproductive.

There is no Summary value for this column.

If the column is consolidated, the same calculation is performed using the consolidated values.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGANET
Internal Field Name	ForActAvlStf2
Description	Forecast group’s Actual Net Staff 2 *Calculation*: Forecast group’s Actual Scheduled Without Adjustments Without Unproductive - Forecast group’s Actual Required Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGANET J

Forecast Group's Actual Net Staff 1 (with Staff Adjustments)

The number of scheduled staff, including any staff adjustments, over or under the actual number of staff required. This value is calculated as:

Forecast Group's Actual Scheduled With Staff Adjustments And Without Unproductive – Forecast Group's Actual Required Without Unproductive

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistic compares the Forecast Group's Actual Scheduled With Staff Adjustments to the Forecast Group's Actual Required Without Unproductive.

There is no Summary value for this column.

If the column is consolidated, the same calculation is performed using the consolidated values.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGANET J
Internal Field Name	ForActAvlStf1
Description	Forecast group’s Actual Net Staff 1 *Calculation*: Forecast group’s Actual Scheduled With Adjustments Without Unproductive - Forecast group’s Actual Required Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGANETC

Forecast Group’s Actual Net Calls

The estimated number of calls that must be outsourced from the forecast group to achieve a net staff of zero for the volume-adjustable staff groups in the routing set.

This calculation is based on actual calls offered, actual AHT, and revised scheduled without adjustments for the staff groups in the routing set. The result may depend on staff levels for staff groups not in the selected set of staff groups.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGANETC

Internal Field Name	ForActNetCalls
Description	Forecast group's Actual Net Calls The estimated number of calls that must be outsourced from the forecast group to achieve a net staff of zero for the volume-adjustable staff groups in the routing set. *Calculation*: An estimate based on actual calls offered, actual AHT, and revised scheduled without adjustments for the staff groups in the routing set. The result may depend on staff levels for staff groups not in the selected set of staff groups.
Min. Decimals	0
Max. Decimals	2
Min. Value	-999,999,999
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGANETC J

Forecast Group's Actual Net Calls (with Staff Adjustments)

The estimated number of calls that must be outsourced from the forecast group to achieve a net staff of zero for the volume-adjustable staff groups in the routing set.

This calculation is based on actual calls offered, actual AHT, revised service level goals, and revised scheduled with adjustments for the staff groups in the routing set. The result may depend on staff levels for staff groups not in the selected set of staff groups.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGANETC J
Internal Field Name	ForActAdjNetCalls
Description	Forecast group's Actual Net Calls With Adjustments The estimated number of calls that must be outsourced from the forecast group to achieve a net staff of zero for the volume-adjustable staff groups in the routing set. *Calculation*: An estimate based on actual calls offered, actual AHT, revised service level goals, and revised scheduled with adjustments for the staff groups in the routing set. The result may depend on staff levels for staff groups not in the selected set of staff groups.
Min. Decimals	0
Max. Decimals	2

Min. Value	-999,999,999
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGAREQ

Forecast Group's Actual Required without Unproductive

The number of staff required based on actual contacts and AHT without unproductive time included in the calculation.

This value is calculated based on: Actual NCO, Actual AHT, the Tendency to Abandon, and Service Quality Goal.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

If the staffing basis is Workload, this value is calculated as:

(Actual Number of Contacts Offered * Actual AHT * (1/60)* (1/Forecast Unit))

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGAREQ
Internal Field Name	ForActReqStfWoUnp
Description	Forecast group’s Actual Required Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGAREQC

Forecast Group’s Actual Required Calls

The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero for each volume-adjustable staff group in the routing set. These are the calls “required” to achieve zero net staff.

This calculation is based on Actual Contacts Offered, Actual AHT, Forecast Group's Actual Scheduled without Staff Adjustments and without Unproductive, and service level goals. The result may depend on staff levels for staff-adjustable staff groups in the routing set.

The Summary value is a total. In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGAREQC
Internal Field Name	
Description	Forecast group’s Actual Required Calls The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero for each volume-adjustable staff group in the routing set. These are the calls “required” to achieve zero net staff. *Calculation*: An estimate based on Actual Contacts Offered, Actual AHT, Forecast Group's Actual Scheduled without Staff Adjustments and without Unproductive, and service level goals. The result may depend on staff levels for staff-adjustable staff groups in the routing set.
Min. Decimals	
Max. Decimals	
Min. Value	
Max. Value	
Table Name	
Field Name	

FGAREQC J

Forecast Group’s Actual Required Calls with Staff Adjustments and without Unproductive

The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero for each volume-adjustable staff group in the routing set. These are the calls “required” to achieve zero net staff.

This calculation is based on Actual Contacts Offered, Actual AHT, Forecast Group's Actual Scheduled with Staff Adjustments, and service level goals. The result may depend on staff levels for staff-adjustable staff groups in the routing set.

The Summary value is a total. In consolidated mode, this value is the sum of the individual values for each period.

FGASCH

Forecast Group's Actual Scheduled without Staff Adjustments and Unproductive

In a single-skill configuration, this is the current number of agents that are scheduled to be working, based only on their official schedules and not including staff adjustments or unproductive time. This value is the same as FGRSCH in a single skill configuration, because the number of staff actually scheduled and scheduled are the same.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

The single-skill calculation for this column is as follows:

Staff Group's Revised Scheduled Without Staff Adjustments * (1 - Unproductive Shrinkage Percentage)

In a multiskill configuration, this value is calculated by the skill-based calculation process using: Actual NCO, Actual AHT, and Staff Group's Revised Scheduled Without Staff Adjustments.

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, this column is blank, if the forecast group has children.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGASCH
Internal Field Name	ForActSchStfWoAdj
Description	Forecast group’s Actual Scheduled Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGASCH J

Forecast Group's Actual Scheduled with Staff Adjustments and without Unproductive

In a single-skill configuration, this is the current number of agents that are scheduled to work and this value includes staff adjustments and not unproductive time. This value is the same as FGRSCH J in a single skill configuration, because the number of staff actually scheduled and scheduled are the same.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

The single-skill calculation for this column is as follows:

Staff Group's Revised Scheduled With Staff Adjustments * (1 - Unproductive Shrinkage Percentage)

In a multiskill configuration, this value is calculated by the skill-based calculation process using Actual NCO, Actual AHT, and Staff Group's Revised Scheduled With Staff Adjustments.

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration this column is blank, if the forecast group has children.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGASCH J
Internal Field Name	ForActSchStfWAdj
Description	Forecast group’s Actual Scheduled With Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGLOCK

Forecast Group's ACD Data Lock Indicator

This column contains a check mark if the forecast group's ACD statistics are locked for that period. Locked data can be manually edited, but it cannot be refreshed (through the ACD interface or through the aggregation of data from child forecast groups). When you edit ACD data such as Actual NCO, Actual AHT, and Actual NCH, that period is automatically locked and a check mark is displayed.

Default Title Name	FGLOCK
Internal Field Name	ForAutoLockoutFlag
Description	Forecast group’s ACD data lock indicator
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	ITD_FOR_CALL_STAT
Field Name	AUTO_LOCKOUT_FLAG

FGNOTE

Forecast Group Note

This column displays the user-supplied note for the forecast group and time period.

Default Title Name	FGNOTE
Internal Field Name	ForNote
Description	Forecast group note
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	ITD_FOR_CALL_NOTE
Field Name	NOTE

FGSTATUS

Forecast Group's ACD Data Status Indicator

This column displays the status of the ACD data for the forecast group and time period. There are four possible values:

Complete: All ACD data has been received for a given period for a particular forecast or staff group.

Incomplete: Indicates that the data for one or more data groups of the forecast or staff group is missing for that period.

Unavailable: Indicates that no data has been received for the forecast or staff group for that period.

Manual edit: The user manually changed the data.

Default Title Name	FGSTATUS
Internal Field Name	ForStatus
Description	Forecast group’s ACD data status indicator
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	ITD_FOR_CALL_STAT
Field Name	STATUS

FGONET

Forecast Group's Original Net Staff

The original number of scheduled staff over or under the number originally required. This value is calculated as:

Forecast Group's Original Scheduled Without Unproductive - Forecast Group's Original Required Without Unproductive

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistic compares the Forecast Group's Original Scheduled to the Forecast Group's Original Required Without Unproductive.

There is no Summary value for this column.

If the column is consolidated, the same calculation is performed using the consolidated values.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGONET
Internal Field Name	ForOrgAvlStf
Description	Forecast group’s Original Net Staff *Calculation*: Forecast group’s Original Scheduled Without Unproductive – Forecast group’s Original Required Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGONETC

Forecast Group’s Original Net Calls

The estimated number of calls that must be outsourced from a forecast group to achieve a net staff of zero for all volume-adjustable staff groups in a routing set.

This calculation is based on original forecast volume, original forecast AHT, original scheduled staff, and service level goals. The result may depend on staff levels for staff-adjustable staff groups in the routing set.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGONETC

Internal Field Name	ForOrgNetCalls
Description	Forecast group's Original Net Calls The estimated number of calls that must be outsourced from a forecast group to achieve a net staff of zero for all volume-adjustable staff groups in a routing set. *Calculation*: An estimate based on original forecast volume, original forecast AHT, original scheduled staff, and service level goals. The result may depend on staff levels for staff-adjustable staff groups in the routing set.
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGOREQ

Forecast Group's Original Required without Unproductive

The number of staff required based on the original forecast without unproductive time included in the calculation.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

This value is calculated based on: Original Forecast Volume, Original Forecast AHT, Tendency to Abandon, and the Service Quality Goal.

If the staffing basis is Workload, this value is calculated as:

(Original Forecast Volume * Original Forecast AHT * (1/60)* (1/Forecast Unit))

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGOREQ
Internal Field Name	ForOrgReqStfWoUnp
Description	Forecast group's Original Required Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0

Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGOREQC

Forecast Groups’s Original Required Calls

The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero for each volume-adjustable staff group in the routing set. These are the calls “required” to achieve zero net staff.

This calculation is based on original forecast volume, original AHT, original service goals, and original scheduled staff. The result may depend on staff levels for the staff-adjustable staff groups in the routing set.

The Summary value is a total. In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGOREQC
Internal Field Name	ForOrgReqCalls
Description	Forecast group’s Original Required Calls The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero for each volume-adjustable staff group in the routing set. These are the calls “required” to achieve zero net staff. *Calculation*: An estimate based on original forecast volume, original AHT, original service goals, and original scheduled staff. The result may depend on staff levels for the staff-adjustable staff groups in the routing set.
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGOSCH

Forecast Group's Original Scheduled without Unproductive

The number of agents scheduled at the time this intra-day performance forecast was created, with unproductive time removed. In a single- skill configuration, this column is calculated as follows:

Staff Group's Original Scheduled * (1 - Unproductive Shrinkage Percentage/100)

In a multiskill configuration, this value is calculated by the skill- based calculation process based on: Original Forecast Volume, Original Forecast AHT, and Staff Group's Revised Scheduled Without Staff Adjustments.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGOSCH
Internal Field Name	ForOrgSchStf
Description	Forecast group’s Original Scheduled Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRNET

Forecast Group's Revised Net Staff 2

The number of scheduled staff over or under the revised number of staff required. The value is calculated as:

Forecast Group's Revised Scheduled Without Staff Adjustments And Unproductive – Forecast Group's Revised Required Without Unproductive

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistic compares the Forecast Group's Revised Scheduled Without Staff Adjustments to the Forecast Group's Revised Required Without Unproductive.

There is no Summary value for this column.

If the column is consolidated, the same calculation is performed using the consolidated values.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGRNET
Internal Field Name	ForRvsAvlStf2

Description	Forecast group’s Revised Net Staff 2 *Calculation*: Forecast group’s Revised Scheduled Without Adjustments Without Unproductive – Forecast group’s Revised Required Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRNET J

Forecast Group's Revised Net Staff 1 (with Staff Adjustments)

The number of scheduled staff, including adjustments, over or under the revised number of staff required. This value is calculated as:

Forecast Group's Revised Scheduled With Staff Adjustments And Without Unproductive – Forecast Group's Revised Required Without Unproductive

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistic compares the Forecast Group's Revised Scheduled With Staff Adjustments to the Forecast Group's Revised Required Without Unproductive.

There is no Summary value for this column.

If the column is consolidated, the same calculation is performed using the consolidated values.

In an Allocate contact allocation configuration this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGRNET J
Internal Field Name	ForRvsAvlStf1
Description	Forecast group’s Revised Net Staff 1 *Calculation*: Forecast group’s Revised Scheduled With Adjustments Without Unproductive – Forecast group’s Revised Required Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99

Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRNETC

Forecast Group's Revised Net Calls

The estimated number of calls that must be outsourced from the forecast group to achieve a net staff of zero for the volume-adjustable staff groups in the routing set.

This calculation is based on revised forecast volume, revised forecast AHT, and revised scheduled without adjustments for the staff groups in the routing set. The result may depend on staff levels of the staff-adjustable staff groups in the routing set.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGRNETC
Internal Field Name	ForRvsNetCalls
Description	Forecast group's Revised Net Calls The estimated number of calls that must be outsourced from the forecast group to achieve a net staff of zero for the volume-adjustable staff groups in the routing set. *Calculation*: An estimate based on revised forecast volume, revised forecast AHT, and revised scheduled without adjustments for the staff groups in the routing set. The result may depend on staff levels of the staff-adjustable staff groups in the routing set.
Min. Decimals	0
Max. Decimals	2
Min. Value	-999,999,999
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRNETC J

Forecast Group's Revised Net Calls (with Staff Adjustments)

The estimated number of calls that must be outsourced from the forecast group to achieve a net staff of zero for the volume-adjustable staff groups in the routing set.

This calculation is based on revised forecast volume, revised forecast AHT, and revised scheduled with adjustments for the staff groups in the routing set. The result may depend on revised staff levels for staff-adjustable staff groups in the routing set.

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGRNETC J
Internal Field Name	ForRvsAdjNetCalls
Description	Forecast group's Revised Net Calls The estimated number of calls that must be outsourced from the forecast group to achieve a net staff of zero for the volume-adjustable staff groups in the routing set. *Calculation*: An estimate based on revised forecast volume, revised forecast AHT, and revised scheduled with adjustments for the staff groups in the routing set. The result may depend on revised staff levels for staff-adjustable staff groups in the routing set.
Min. Decimals	0
Max. Decimals	2
Min. Value	-999,999,999
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRREQ

Forecast Group's Revised Required without Unproductive

The revised number of staff required without unproductive time included in the calculation.

This value is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Service Quality Goal.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

If the staffing basis is Workload, this value is calculated as:

(Revised Forecast Volume * Revised Forecast AHT * (1/60)* (1/Forecast Unit))

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGRREQ
Internal Field Name	ForRvsReqStfWoUnp
Description	Forecast group's Revised Required Without Unproductive
Min. Decimals	0
Max. Decimals	2

Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRREQC

Forecast Group's Revised Required Calls

The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero in each volume-adjustable staff group in the routing set. These are the calls "required" to achieve zero net staff in the volume-adjustable staff groups.

This calculation is based on revised forecast volume, revised AHT, and revised scheduled staff without adjustments. The result may depend on revised staff levels for the staff-adjustable staff groups in the routing set.

The Summary value is a total. In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGRREQC
Internal Field Name	ForRvsReqCalls
Description	Forecast group's Revised Required Calls The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero in each volume-adjustable staff group in the routing set. These are the calls "required" to achieve zero net staff in the volume-adjustable staff groups. *Calculation*: An estimate based on revised forecast volume, revised AHT, and revised scheduled staff without adjustments. The result may depend on revised staff levels for the staff-adjustable staff groups in the routing set.
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRREQC J

Forecast Group’s Revised Required Calls (With Staff Adjustments)

The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero in each volume-adjustable staff group in the routing set.

This calculation is based on revised forecasted volume, revised AHT, and revised scheduled staff with adjustments. The result may depend on revised staff levels for the staff-adjustable staff groups in the routing set.

The Summary value is a total. In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	FGRREQC J
Internal Field Name	ForRvsReqAdjCalls
Description	Forecast group’s Revised Required Calls With Adjustments The estimated number of calls that must be offered from the forecast group to achieve a net staff of zero in each volume-adjustable staff group in the routing set. *Calculation*: An estimate revised forecasted volume, revised AHT, and revised scheduled staff with adjustments. The result may depend on revised staff levels for the staff-adjustable staff groups in the routing set.
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRSCH

Forecast Group's Revised Scheduled without Staff Adjustments and Unproductive

The number of agents who are scheduled to work based on the latest schedule tally that does not include staff adjustments or unproductive time. In a single-skill configuration, this column is calculated as follows:

Staff Group's Revised Scheduled Without Staff Adjustments * (1 - Unproductive Shrinkage Percentage/100)

In a multiskill configuration, this value is calculated by the skill-based calculation process based on: Revised Forecast Volume, Revised Forecast AHT, Staff Group's Revised Scheduled Without Staff Adjustments.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGRSCH
Internal Field Name	ForRvsSchStfWoAdj
Description	Forecast group's Revised Scheduled Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

FGRSCH J

Forecast Group's Revised Scheduled with Staff Adjustments and without Unproductive

The number of agents that are scheduled to work based on the latest schedule tally that does include staff adjustments but without unproductive time. In a single-skill configuration, this column is calculated as follows:

Staff Group's Revised Scheduled With Staff Adjustments * (1 - Unproductive Shrinkage Percentage/100)

In a multiskill configuration, this value is calculated by the skill-based calculation process based on: Revised Forecast Volume, Revised Forecast AHT, Staff Group's Revised Scheduled With Staff Adjustments.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	FGRSCH J
Internal Field Name	ForRvsSchStfWAdj
Description	Forecast group's Revised Scheduled With Adjustments Without Unproductive

Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OAAHTDEV

Original vs. Actual AHT Deviation

The amount by which the actual AHT deviates from the original forecast AHT, calculated as:

Actual AHT - Original Forecast AHT

In single-skill contact allocation configurations this column is blank for parent forecast groups.

Default Title Name	OAAHTDEV
Internal Field Name	OrgActAHTNetDev
Description	Original AHT Net Deviation (Actual AHT - Original Forecast AHT)
Min. Decimals	0
Max. Decimals	0
Min. Value	-99,999,999
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OAAHTPDEV

Original vs. Actual AHT Percent Deviation

The percentage by which the actual AHT deviates from the original forecast AHT, calculated as:

(Actual AHT - Original Forecast AHT) / Original Forecast AHT

The Summary value is an average weighted by the Original Forecast Volume.

In single-skill contact allocation configurations this column is blank for parent forecast groups.

In consolidated mode, the same calculation is performed using the consolidated actual and forecast values.

Default Title Name	OAAHTPDEV

Internal Field Name	OrgActAHTPctDev
Description	Original AHT Percent Deviation ([Actual AHT - Original Forecast AHT] / Original Forecast AHT)
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OAAHTRTIO

Original vs. Actual AHT Ratio

The ratio of actual AHT to the original forecast AHT, calculated as:

Actual AHT / Original Forecast AHT

The Summary value is an average weighted by the Original Forecast Volume.

In single-skill contact allocation configurations this column is blank for parent forecast groups.

In consolidated mode, the same calculation is performed using the consolidated actual and forecast values.

Default Title Name	OAAHTRTIO
Internal Field Name	OrgActAHTRatio
Description	Original AHT Ratio (Actual AHT / Original Forecast AHT)
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OABD

Original Forecast Calculated Abandoned

The original estimate of the number of contacts received by the ACD but disconnected by the customer before an agent answers. This field is calculated based on: Original Forecast Volume, Original Forecast AHT, Original Scheduled Staff, and Tendency to Abandon. The Summary value is a total.

For a parent forecast group in an Allocate contact allocation configuration, this value is the sum of the values for the child groups.

In consolidated mode, this value is the sum of the individual values for each period.

This column is blank if the staffing basis is Workload.

Default Title Name	OABD
Internal Field Name	OrgForAbandoned
Description	Original Forecast Abandoned (based on Original Forecast Volume and forecast group's Original Scheduled Without Unproductive)
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OAHT Original Forecast AHT

The estimated amount of time an agent is occupied with incoming work (contacts, paperwork, etc.). AHT is the sum of talk time and wrap-up time. The Summary value is an average weighted by Original Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Original Forecast Volume.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups. AHT is only available for child forecast groups that are routed to a staff group.

Default Title Name	OAHT
Internal Field Name	OrgForAHT
Description	Original Forecast AHT
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999,999.99
Table Name	IDP_FOR_GRP_DET
Field Name	ORG_FOR_AHT

OAVOLDEV Original vs. Actual Volume Deviation

The amount by which the actual contacts offered deviates from the original forecast volume, calculated as:

Actual Contacts Offered - Original Forecast Volume

Default Title Name	OAVOLDEV
Internal Field Name	OrgVolNetDev
Description	Original Volume Net Deviation (Actual Calls Offered - Original Forecast Volume)
Min. Decimals	0
Max. Decimals	0
Min. Value	-99,999,999
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OAVOLPDEV

Original vs. Actual Volume Percent Deviation

The percentage by which the actual contacts offered deviates from the original forecast volume, calculated as:

(Actual Contacts Offered - Original Forecast Volume) / Original Forecast Volume

The Summary value is an average weighted by the Original Forecast Volume.

In consolidated mode, the same calculation is performed using the consolidated actual and forecast values.

Default Title Name	OAVOLPDEV
Internal Field Name	OrgVolPctDev
Description	Original Volume Percent Deviation ([Actual Calls Offered - Original Forecast Volume] / Original Forecast Volume)
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OAVOLRTIO

Original vs. Actual Volume Ratio

The ratio of actual contacts offered to the original forecast volume, calculated as:

Actual Contacts Offered / Original Forecast Volume

The Summary value is an average weighted by the Original Forecast Volume.

In consolidated mode, the same calculation is performed using the consolidated actual and forecast values.

Default Title Name	OAVOLRTIO
Internal Field Name	OrgActVolRatio
Description	Original Volume Ratio (Actual Calls Offered / Original Forecast Volume)
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OBCKLOG

Original Forecast Calculated Backlog

The original estimate of the number of contacts remaining in queue at the end of the period. The Summary value is a simple average per period, as is the partial-day summary value. In consolidated mode, the value for each period is a total for the consolidated groups, and the simple averages are calculated from these consolidated values.

This statistic is available only for forecast groups in a multiskill routing set or forecast groups configured for long service delays. It is calculated only when you use the Calculate Statistics option.

In an Allocate contact allocation configuration, a parent group's value is the sum of the calculated values for its child groups.

ODELAYHR

Original Calculated Average Delay: Hours

The estimated average delay between the time a contact is received by the ACD and the time it is answered by an agent, in hours. This column is calculated using the following: Original Forecast Volume, Original Forecast AHT, Tendency to Abandon, and Forecast Group's Original Scheduled Staff Without Unproductive. The Summary value is an average weighted by Original Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Original Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ODELAYHR
Internal Field Name	OrgAvgDelayHour
Description	Original Calculated Average Delay Hour (based on Original Forecast Volume and the forecast group's Simulated Original Scheduled Without Unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	48
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ODELAYMIN

Original Calculated Average Delay: Minutes

The estimated average delay between the time a contact is received by the ACD and the time it is answered by an agent, in minutes. This column is calculated using the following: Original Forecast Volume, Original Forecast AHT, Tendency to Abandon, and Forecast Group's Original Scheduled Staff Without Unproductive. The Summary value is an average weighted by Original Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Original Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ODELAYMIN
Internal Field Name	OrgAvgDelayMinute
Description	Original Calculated Average Delay (based on Original Forecast Volume and the forecast group's Simulated Original Scheduled Without Unproductive)
Min. Decimals	0
Max. Decimals	1

Min. Value	0
Max. Value	2880.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ODELAYSEC

Original Calculated Average Delay: Seconds

The estimated average delay between the time a contact is received by the ACD and the time it is answered by an agent, in seconds. This column is calculated using the following: Original Forecast Volume, Original Forecast AHT, Tendency to Abandon, and Forecast Group's Original Scheduled Staff Without Unproductive. The Summary value is an average weighted by Original Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Original Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	ODELAYSEC
Internal Field Name	OrgAvgDelaySec
Description	Original Calculated Average Delay (based on Original Forecast Volume and forecast group’s Simulated Original Scheduled Without Unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	172800.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

Original Simulated Queue Backlog

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	TSimForOrgBacklog
Description	Forecast group’s Simulated Original Queue Backlog

Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	9,999,999
Table Name	NA
Field Name	NA

OSL

Original Calculated Service Level Percent

The estimated number of offered contacts that can be answered within the acceptable delay (SLGOALSEC, SLGOALMIN, or SLGOALHR). This column is calculated based on: Original Forecast Volume, Original Forecast AHT, Tendency to Abandon, Service Level Goal (in seconds, minutes, or hours) and Original Scheduled Staff Without Unproductive. The Summary value is an average weighted by Original Forecast Volume.

In consolidated mode, this value is an average weighted by the Original Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	OSL
Internal Field Name	OrgSrvcLevPct
Description	Original Calculated Service Level Percent (based on Original Forecast Volume and forecast group's Simulated Original Scheduled Without Unproductive)
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OVOL

Original Forecast Volume

The estimated volume of offered work (contacts, paperwork, etc.). The Summary value is a total.

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, child forecast group values are filled through allocation.

Default Title Name	OVOL
Internal Field Name	OrgForVol
Description	Original Forecast Volume
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	IDP_FOR_GRP_DET
Field Name	ORG_FOR_VOL

RAAHTDEV

Revised vs. Actual AHT Deviation

The amount by which the actual AHT deviates from the revised forecast AHT, calculated as:

Actual AHT - Revised Forecast AHT

In single-skill contact allocation configurations this column is blank for parent forecast groups.

Default Title Name	RAAHTDEV
Internal Field Name	RvsActAHTNetDev
Description	Revised AHT Net Deviation (Actual AHT - Revised Forecast AHT)
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RAAHTPDEV

Revised vs. Actual AHT Percent Deviation

The percentage by which the actual AHT deviates from the revised forecast AHT, calculated as:

(Actual AHT - Revised Forecast AHT) / Revised Forecast AHT

The Summary value is an average weighted by the Revised Forecast Volume.

In single-skill contact allocation configurations this column is blank for parent forecast groups.

In consolidated mode, the same calculation is performed using the consolidated actual and forecast values.

Default Title Name	RAAHTPDEV
Internal Field Name	RvsAHTPctDev
Description	Revised AHT Percent Deviation ([Actual AHT - Original Forecast AHT] / Original Forecast AHT)
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RAAHTRTIO

Revised vs. Actual AHT Ratio

The ratio of actual AHT to the revised forecast AHT, calculated as:

Actual AHT / Revised Forecast AHT

The Summary value is an average weighted by the Revised Forecast Volume.

In single-skill contact allocation configurations this column is blank for parent forecast groups.

In consolidated mode, the same calculation is performed using the consolidated actual and forecast values.

Default Title Name	RAAHTRTIO
Internal Field Name	RvsActAHTRatio
Description	Revised AHT Ratio (Actual AHT / Original Forecast AHT)
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RABD

Revised Forecast Calculated Abandoned without Staff Adjustments and Unproductive

The revised estimate of the number of contacts received by the ACD but disconnected by the customer before an agent answers. This revised estimate does not include staff adjustments. This field is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Revised Scheduled Without Staff Adjustments And Unproductive.

For a parent forecast group in an Allocate call allocation configuration, this value is the sum of the values of the child groups.

In consolidated mode, this value is the sum of the individual values for each period.

This column is blank if the staffing basis is Workload.

Default Title Name	RABD
Internal Field Name	RvsForAbandoned2
Description	Revised Forecast Abandoned (based on Revised Forecast and Forecast Group's Revised Scheduled Without Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RABD J

Revised Forecast Calculated Abandoned with Staff Adjustments and without Unproductive

The revised estimate of the number of contacts received by the ACD but disconnected by the customer before an agent answers. This revised estimate includes any staff adjustments. This field is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Revised Scheduled With Staff Adjustments And Without Unproductive. The Summary value is a total.

For a parent forecast group in an Allocate contact allocation configuration, this value is the sum of the values of the child groups.

In consolidated mode, this value is the sum of the individual values for each period.

This column is blank if the staffing basis is Workload.

Default Title Name	RABD J

Internal Field Name	RvsForAbandoned1
Description	Revised Forecast Abandoned (based on Revised Forecast and forecast group's Revised Scheduled With Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RAHT

Revised Forecast AHT

The revised estimated average handle time. Initially, the Original Forecast AHT value fills this field until changes are made to the forecast through reforecasting or manual editing. The Summary value is an average weighted by the Revised Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Revised Forecast Volume.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups. The forecast AHT is available only for child forecast groups that are routed to a staff group.

Default Title Name	RAHT
Internal Field Name	RvsForAHT
Description	Revised Forecast AHT
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999,999.99
Table Name	IDP_FOR_GRP_DET
Field Name	RVS_FOR_AHT

RAHTTS

Revised Forecast AHT Update Timestamp

The date and time at which the Revised Forecast AHT field was first written or last changed.

Default Title Name	RAHTTS
Internal Field Name	RvsForAhtUpdTS
Description	Revised Forecast AHT Update Timestamp
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	IDP_FOR_GRP_DET
Field Name	RVS_FOR_AHT_UPD_TS

RAVOLDEV

Revised vs. Actual Volume Deviation

The amount by which the actual contacts offered deviates from the revised forecast volume, calculated as:

Actual Contacts Offered - Revised Forecast Volume

Default Title Name	RAVOLDEV
Internal Field Name	RvsActVolNetDev
Description	Revised Volume Net Deviation (Actual Calls Offered - Revised Forecast Volume
Min. Decimals	0
Max. Decimals	0
Min. Value	-99,999,999
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RAVOLPDEV

Revised vs. Actual Volume Percent Deviation

The percentage by which the actual contacts offered deviates from the revised forecast volume, calculated as:

(Actual Contacts Offered - Revised Forecast Volume) / Revised Forecast Volume

The Summary value is an average weighted by the Revised Forecast Volume.

In consolidated mode, the same calculation is performed using the consolidated actual and forecast values.

Default Title Name	RAVOLPDEV
Internal Field Name	RvsActVolPctDev
Description	Revised Volume Percent Deviation ([Actual Calls Offered - Revised Forecast Volume] / Revised Forecast Volume)
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RAVOLRTIO

Revised vs. Actual Volume Ratio

The ratio of actual contacts offered to the revised forecast volume, calculated as:

Actual Contacts Offered / Revised Forecast Volume

The Summary value is an average weighted by the Revised Forecast Volume.

In consolidated mode, the same calculation is performed using the consolidated actual and forecast values.

Default Title Name	RAVOLRTIO
Internal Field Name	RvsActVolRatio
Description	Revised Volume Ratio (Actual Calls Offered / Revised Forecast Volume)
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RBCKLOG

Revised Forecast Calculated Backlog

The revised estimate of the number of contacts remaining in queue at the end of the period. The Summary value is a simple average per period, as is the partial-day summary value. In consolidated mode, the value for each period is a total for the consolidated groups, and the simple averages are calculated from these consolidated values.

This statistic is available only for forecast groups in a multiskill routing set or forecast groups configured for long service delays. It is calculated only when you use the Calculate Statistics option.

In an Allocate contact allocation configuration, a parent group's value is the sum of the calculated values for its child groups.

RBCKLOG J

Revised Forecast Calculated Backlog with Staff Adjustments

The revised estimate of the number of contacts remaining in queue at the end of the period, with staff adjustments taken into account. The Summary value is a simple average per period, as is the partial-day summary value. In consolidated mode, the value for each period is a total for the consolidated groups, and the simple averages are calculated from these consolidated values.

This statistic is available only for forecast groups in a multiskill routing set or forecast groups configured for long service delays. It is calculated only when you use the Calculate Statistics option.

In an Allocate contact allocation configuration, a parent group's value is the sum of the calculated values for its child groups.

RDELAY HR

Revised Calculated Average Delay 2: Hours

The revised estimate of the average delay between the time a contact is received by the ACD and the time it is answered by an agent, in hours. This column is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Forecast Group's Revised Scheduled Without Staff Adjustments And Unproductive. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Revised Forecast Volume. If one of the forecast groups uses Service Level as the service quality goal, this value is converted to an average delay to be consolidated with the other group.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	RDELAY HR
Internal Field Name	RvsCalcAvgDelay2Hr
Description	Revised Calculated Average Delay 2 Hour (based on Revised Forecast Volume and the forecast group's

	Simulated Revised Scheduled Without Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	48.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RDELAY J HR

Revised Calculated Average Delay 1 (with Staff Adjustments): Hours

The revised estimate of the average delay between the time a contact is received by the ACD and the time it is answered by an agent, in hours. The column is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Forecast Group's Revised Scheduled With Staff Adjustments And Without Unproductive. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Revised Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

RDELAY J MIN

Revised Calculated Average Delay 1 (with Staff Adjustments): Minutes

The revised estimate of the average delay between the time a contact is received by the ACD and the time it is answered by an agent, in minutes. The column is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Forecast Group's Revised Scheduled With Staff Adjustments And Without Unproductive. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Revised Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	RDELAY J MIN
Internal Field Name	RvsCalcAvgDelay1Minute
Description	Revised Calculated Average Delay 1 (based on Revised Forecast Volume and the forecast group's Simulated Revised Scheduled With Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	48.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RDELAY J SEC

Revised Calculated Average Delay 1 (with Staff Adjustments): Seconds

The revised estimate of the average delay between the time a contact is received by the ACD and the time it is answered by an agent, in seconds. The column is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Forecast Group's Revised Scheduled With Staff Adjustments And Without Unproductive. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Revised Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	RDELAY J SEC
Internal Field Name	RvsCalcAvgDelay1
Description	Revised Calculated Average Delay 1 (based on Revised Forecast Volume and the forecast group's Simulated Revised Scheduled With Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	172800.0

Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RDELAY MIN

Revised Calculated Average Delay 2: Minutes

The revised estimate of the average delay between the time a contact is received by the ACD and the time it is answered by an agent, in minutes. This column is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Forecast Group's Revised Scheduled Without Staff Adjustments And Unproductive. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Revised Forecast Volume. If one of the forecast groups uses Service Level as the service quality goal, this value is converted to an average delay to be consolidated with the other group.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	RDELAY MIN
Internal Field Name	RvsCalcAvgDelay2Minute
Description	Revised Calculated Average Delay 2 Minute (based on Revised Forecast Volume and the forecast group's Simulated Revised Scheduled Without Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	2880.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RDELAY SEC

Revised Calculated Average Delay 2: Seconds

The revised estimate of the average delay between the time a contact is received by the ACD and the time it is answered by an agent, in seconds. This column is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, and Forecast Group's Revised Scheduled Without Staff Adjustments And Unproductive. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is an average weighted by each forecast group's Revised Forecast Volume. If one of the forecast groups uses Service Level as the service quality goal, this value is converted to an average delay to be consolidated with the other group.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	RDELAY SEC
Internal Field Name	RvsCalcAvgDelay2
Description	Revised Calculated Average Delay 2 (based on Revised Forecast Volume and the forecast group's Simulated Revised Scheduled Without Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	1
Min. Value	0
Max. Value	172800.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

Revised Forecast Handling Capability Without Adjustments

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	RvsVolHanCapWoAdj
Description	Revised Volume Handling Capability Without Adjustments
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

Revised Simulated Queue Backlog With Adjustments

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	TSimForRvsBacklog1
Description	Forecast group's Simulated Revised Queue Backlog With Adjustments
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	9,999,999
Table Name	NA
Field Name	NA

Revised Simulated Queue Backlog Without Adjustments

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	TSimForRvsBacklog2
Description	Forecast group's Simulated Revised Queue Backlog Without Adjustments
Min. Decimals	0
Max. Decimals	0
Min. Value	0
Max. Value	9,999,999
Table Name	NA
Field Name	NA

Revised Volume Handling Capability With Adjustments

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	RvsVolHanCapWAdj
Description	Revised Volume Handling Capability With Adjustments
Min. Decimals	0
Max. Decimals	0

Min. Value	0
Max. Value	999,999,999
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RSL

Revised Calculated Service Level Percent 2

The revised estimate of the percentage of offered contacts answered within the acceptable delay (SLGOALSEC, SLGOALMIN, or SLGOALHR). This column is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon, the Service Level Goal (in seconds, minutes, or hours), and Forecast Group's Revised Scheduled Without Staff Adjustments And Unproductive. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is an average weighted by Revised Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	RSL
Internal Field Name	RvsCalcSrvcLevPct2
Description	Revised Calculated Service Level Percent 2 (based on Revised Forecast and the forecast group's Simulated Revised Scheduled Without Adjustments Without unproductive)
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RSL J

Revised Calculated Service Level Percent 1 (with Staff Adjustments)

The revised estimate of the percentage of offered contacts answered within the acceptable delay (SLGOALSEC, SLGOALMIN, or SLGOALHR). Staff Adjustments are included in the calculation. This column is calculated based on: Revised Forecast Volume, Revised Forecast AHT, Tendency to Abandon,

the Service Level Goal (in seconds, minutes, or hours), and Revised Scheduled With Staff Adjustments And Without Unproductive. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, the value is an average weighted by Revised Forecast Volume.

This column is blank:

- For parent forecast groups in an Allocate contact allocation configuration.
- If the staffing basis is Workload.

Default Title Name	RSL J
Internal Field Name	RvsCalcSrvcLevPct1
Description	Revised Calculated Service Level Percent 1 (based on Revised Forecast and the forecast group's Simulated Revised Scheduled With Adjustments Without Unproductive)
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	100.00
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RVOL

Revised Forecast Volume

The revised estimated volume of work (contacts, paperwork, etc.). The Revised Forecast Volume displays the Original Forecast Volume until any kind of change is made to the original forecast. The Summary value is a total.

In consolidated mode, this value is the sum of the individual values for each period.

In an Allocate contact allocation configuration, the Revised Forecast Volume for the child forecast groups is initially filled through allocation. Reforecasts, manual changes, and the restoration of the original forecast are not automatically allocated to child forecast groups. You must manually reallocate these changes.

Default Title Name	RVOL
Internal Field Name	RvsForVol
Description	Revised Forecast Volume
Min. Decimals	0
Max. Decimals	0
Min. Value	0

Max. Value	999,999,999
Table Name	IDP_FOR_GRP_DET
Field Name	RVS_FOR_VOL

RVOLTS

Revised Forecast Volume Update Timestamp

The date and time at which the Revised Forecast Volume field was first written or last changed.

Default Title Name	RVOLTS
Internal Field Name	RvsForVolUpdTS
Description	Revised Forecast Volume Update Timestamp
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	IDP_FOR_GRP_DET
Field Name	RVS_FOR_VOL_UPD_TS

Simulated Actual Scheduled With Adjustments Without Unproductive

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	SimForActSchStfWAdj
Description	Forecast group’s Simulated Actual Scheduled With Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

Simulated Original Scheduled Without Unproductive

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	SimForOrgSchStf
Description	Forecast group's Simulated Original Scheduled Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

Simulated Revised Scheduled With Adjustments Without Unproductive

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	SimForRvsSchStfWAdj
Description	Forecast group's Simulated Revised Scheduled With Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

Simulated Revised Scheduled Without Adjustments Without Unproductive

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.

Internal Field Name	SimForRvsSchStfWoAdj
Description	Forecast group's Simulated Revised Scheduled Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SL-ACD

Actual Service Level

The service level percentage reported by the ACD (percentage of offered contacts answered within a certain acceptable delay). The Summary value is an average weighted by the Actual Number of Contacts Offered. The acceptable delay is set in the ACD.

In consolidated mode, this value is an average weighted by each forecast group's ANCO.

Default Title Name	SL-ACD
Internal Field Name	SrvcLevPct
Description	Actual Service Level Percent from the ACD.
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	9999.99
Table Name	ITD_FOR_CALL_STAT
Field Name	SRVC_LVL_PCT

SLGOAL

Service Level Goal: Percent

The desired percentage of offered contacts that are answered within a certain acceptable delay: SLGOALSEC, SLGOALMIN, or SLGOALHR. Both values (SLGOAL and the acceptable delay) are specified in the forecasting scenario. The Summary value is an average weighted by Revised Forecast Volume.

In consolidated mode, this value is calculated as follows: First, if any forecast group has Average Delay as its service quality goal, that goal is converted to an equivalent service level goal. Second, all forecast

groups' acceptable-delay value is converted to the Service Level Goal specified for this routing set. Finally, the individual values are weighted by Revised Forecast Volume and averaged.

This column is blank:

For a parent forecast group in an Allocate contact allocation configuration. Service Level goals are defined individually by child group and are not allocated.

- If the staffing basis is Workload.
- If this column is not in consolidated mode and the staffing basis is Average Delay.

Default Title Name	SLGOAL
Internal Field Name	SrvcLevGoalPct
Description	Service Level Goal – Percent
Min. Decimals	0
Max. Decimals	2
Min. Value	00.01
Max. Value	99.99
Table Name	IDP_FOR_GRP_DET
Field Name	SRVC_LVL_GOAL_PCT

SLGOALHR

Service Level Goal: Hours

The number of hours within which the desired percentage of offered contacts (SLGOAL) are answered. Both values (SLGOAL and SLGOALHR) are stored in the forecasting scenario.

In consolidated mode, this value is the service level goal you've specified for this routing set.

This column is blank:

- For a parent forecast group in an Allocate contact allocation configuration. Service level goals are defined individually by child group and are not allocated.
- If the staffing basis is Workload.

This value is displayed even if you have selected Average Delay as the staffing basis, and again, it is stored (if only as a default value) in the forecasting scenario. However, it is invisible in the scenario as long as the staffing basis is Average Delay.

Default Title Name	SLGOALHR
Internal Field Name	SrvcLevGoalHour
Description	Service Level Goal – Hours
Min. Decimals	0
Max. Decimals	1

Min. Value	0
Max. Value	48
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SLGOALMIN

Service Level Goal: Minutes

The number of minutes within which the desired percentage of offered contacts (SLGOAL) are answered. Both values (SLGOAL and SLGOALMIN) are stored in the forecasting scenario.

In consolidated mode, this value is the service level goal you've specified for this routing set.

This column is blank:

- For a parent forecast group in an Allocate contact allocation configuration. Service level goals are defined individually by child group and are not allocated.
- If the staffing basis is Workload.

This value is displayed even if you have selected Average Delay as the staffing basis, and again, it is stored (if only as a default value) in the forecasting scenario. However, it is invisible in the scenario as long as the staffing basis is Average Delay.

Default Title Name	SLGOALMIN
Internal Field Name	SrvcLevGoalMinute
Description	Service Level Goal – Minutes
Min. Decimals	0
Max. Decimals	1
Min. Value	0.0
Max. Value	2880.0
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SLGOALSEC

Service Level Goal: Seconds

The number of seconds within which the desired percentage of offered contacts (SLGOAL) are answered. Both values (SLGOAL and SLGOALSEC) are stored in the forecasting scenario.

In consolidated mode, this value is the service level goal you have specified for this routing set.

This column is blank:

- For a parent forecast group in an Allocate contact allocation configuration. Service level goals are defined individually by child group and are not allocated.
- If the staffing basis is Workload.

This value is displayed even if you have selected Average Delay as the staffing basis, and again, it is stored (if only as a default value) in the forecasting scenario. However, it is invisible in the scenario as long as the staffing basis is Average Delay.

Default Title Name	SLGOALSEC
Internal Field Name	SrvcLevGoalSec
Description	Service Level Goal – Seconds
Min. Decimals	0
Max. Decimals	1
Min. Value	0.1
Max. Value	172800.0
Table Name	IDP_FOR_GRP_DET
Field Name	SRVC_LVL_GOAL_SEC

TENDENCYTOABANDON

Tendency to Abandon

The estimated rate at which customers abandon (disconnect after reaching the ACD but before an agent answers). You specify this value in the forecasting scenario.

In consolidated mode, this value is an average weighted by each forecast group's Revised Forecast Volume.

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

Default Title Name	DROP
Internal Field Name	DropOutRate
Description	Dropout Rate
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	9,999,999.99
Table Name	IDP_FOR_GRP_DET
Field Name	DROPOUT_RATE

WRAP

Wrap-up Seconds

The amount of time an agent spends completing a transaction after the associated contact has already been disconnected. The Summary value is an average weighted by Revised Forecast Volume. In consolidated mode, all forecast group values are weighted by each forecast group's revised forecast volume. In an Allocate contact allocation configuration, this value is blank for all child forecast groups. The Wrap-Up seconds comes from the forecasting scenario.

Default Title Name	WRAP
Internal Field Name	WrapUpSec
Description	Wrap-up Seconds
Min. Decimals	0
Max. Decimals	1
Min. Value	0.0
Max. Value	9,999.9
Table Name	IDP_FOR_GRP_DET
Field Name	WRAP_UP_SEC

Staff Group Statistics

This section provides descriptions of intra-day performance columns that pertain primarily to staff group statistics.

AABSREQADH 1

Absolute Actual Required Adherence 1

(Based on Staff Group's Actual Required with staff adjustments and without Unproductive)

This percentage measures adherence to the actual required staff without regard to whether non-adherence is positive or negative. For example, if 10 agents are required and 12 are actually working, the absolute adherence is 80%. Likewise, if 10 agents are required and only 8 are actually working, the absolute adherence is also 80%. This percentage is thus based on the absolute value of the Actual Required Deviation 1.

The Summary value is an average weighted by the Staff Group's Actual Required With Staff Adjustments And Without Unproductive.

It is calculated as:

(Staff Group's Actual Required With Staff Adjustments And Without Unproductive - | Actual Required Deviation 1 |) / Staff Group's Actual Required With Staff Adjustments And Without Unproductive * 100.

In consolidated mode, the same calculation is performed using the consolidated values.

In a staff allocation configuration this column is blank if the staff group has a parent.

Default Title Name	AABSREQADH1
Internal Field Name	AbsActReqAdher1
Description	Absolute Actual Required Adherence 1 (This statistic is based on Staff group's Actual Required With Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	1
Min. Value	-9999.9
Max. Value	9999.9
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AABSREQADH 2

Absolute Actual Required Adherence 2

(Based on Staff Group's Actual Required without Staff Adjustments and Unproductive)

This percentage measures adherence to the actual required staff without regard to whether the non-adherence is positive or negative. For example, if 10 agents are required and 12 are actually working, the absolute adherence is 80%. Likewise, if 10 agents are required and only 8 are actually working, the absolute adherence is also 80%. This percentage is thus based on the absolute value of Actual Required Deviation 2.

It is calculated as:

(Staff Group's Actual Required Without Staff Adjustments And Unproductive - | Actual Required Deviation 2 |) / Staff Group's Actual Required Without Staff Adjustments And Unproductive * 100.

The Summary value is an average weighted by the Staff Group's Actual Required Without Staff Adjustments And Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

In a staff allocation configuration this column is blank if the staff group has a parent.

Default Title Name	AABSREQADH2
Internal Field Name	AbsActReqAdher2
Description	Absolute Actual Required Adherence 2 (This statistic is based on Staff group's Actual Required Without Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	1
Min. Value	-9999.9

Max. Value	9999.9
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ABSSCHADH 1

Absolute Scheduled Adherence 1

(Based on Staff Group's Original Scheduled)

This percentage measures adherence to the original schedule without regard to whether the non-adherence is positive or negative. For example, if 10 agents are scheduled and 12 are actually working, the absolute adherence is 80%. Likewise, if 10 agents are scheduled and only 8 are actually working, the absolute adherence is also 80%. This percentage is thus based on the absolute value of Scheduled Deviation 1.

It is calculated as:

(Staff Group's Original Scheduled - | Scheduled Deviation 1 |) / Staff Group's Original Scheduled * 100

The Summary value is an average weighted by the Staff Group's Original Scheduled.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	ABSSCHADH1
Internal Field Name	AbsSchAdher1
Description	Absolute Scheduled Adherence 1 (This statistic is based on Staff Group’s Original Scheduled.)
Min. Decimals	0
Max. Decimals	1
Min. Value	-9999.9
Max. Value	9999.9
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ABSSCHADH 2

Absolute Scheduled Adherence 2

(Based on Staff Group's Revised Scheduled with Staff Adjustments)

This percentage measures adherence to the revised scheduled with staff adjustments without regard to whether the non-adherence is positive or negative. For example, if 10 agents are scheduled and 12 are actually working, the absolute adherence is 80%. Likewise, if 10 agents are scheduled and only 8 are actually working, the absolute adherence is also 80%. This percentage is thus based on the absolute value of Scheduled Deviation 2.

It is calculated as:

(Staff Group's Revised Scheduled With Staff Adjustments - | Scheduled Deviation 2 |) / Staff Group's Revised Scheduled With Staff Adjustments * 100

The Summary value is an average weighted by Staff Group's Revised Scheduled With Staff Adjustments.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	ABSSCHADH2
Internal Field Name	AbsSchAdher2
Description	Absolute Scheduled Adherence 2 (This statistic is based on Staff group's Revised Scheduled With Adjustments.)
Min. Decimals	0
Max. Decimals	1
Min. Value	-9999.9
Max. Value	9999.9
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

ABSSCHADH 3

Absolute Scheduled Adherence 3

(Based on Staff Group's Revised Scheduled without Staff Adjustments)

This percentage measures adherence to the revised scheduled staff without staff adjustments without regard to whether the non-adherence is positive or negative. For example, if 10 agents are scheduled and 12 are actually working, the absolute adherence is 80%. Likewise, if 10 agents are scheduled and only 8 are actually working, the absolute adherence is also 80%. This percentage is thus based on the absolute value of Scheduled Deviation 3.

It is calculated as:

(Staff Group's Revised Scheduled Without Staff Adjustments - | Scheduled Deviation 3 |) / Staff Group's Revised Scheduled Without Staff Adjustments * 100

The Summary value is an average weighted by the Staff Group's Revised Scheduled Without Staff Adjustments.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	ABSSCHADH3
Internal Field Name	AbsSchAdher3
Description	Absolute Scheduled Adherence 3 (This statistic is based on Staff group's Revised Scheduled Without Adjustments.)

Min. Decimals	0
Max. Decimals	1
Min. Value	-9999.9
Max. Value	9999.9
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AGGFGRSCH

Forecast Group's Revised Aggregated Scheduled without Staff Adjustments and Unproductive

The number of agents who are scheduled to handle work for this forecast group, based on the latest schedule tally that does not include staff adjustments or unproductive time.

In a single-skill configuration, the value in this column is copied from the Staff Group's Revised Scheduled without Staff Adjustments (SGRSCH) column for the staff group associated with this forecast group.

In a multiskill configuration, this value is calculated by the skill-based calculation process. The calculation is the sum of the following value for all staff groups that can receive work from this forecast group:

Staff Group's Revised Scheduled without Staff Adjustments (SGRSCH) multiplied by the fraction of the total workload handled by the staff group that comes from this forecast group.

To get the total number of staff hours, divide the column’s summary value by two (for 30-minute periods) or by four (for 15-minute periods).

In consolidated mode, the AGGFGRSCH value is the sum of the individual forecast group values for each period.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In an Allocate contact allocation configuration, this column is blank for parent forecast groups.

APS

Actual Positions Staffed

The number of work positions that are actually occupied by agents as reported by the ACD.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	APS
Internal Field Name	ActPosStf
Description	Actual Positions Staffed

Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	ITD_STF_CALL_STAT
Field Name	APS

AREQADH1

Actual Required Adherence 1

(Based on Staff Group's Actual Required with Staff Adjustments and without Unproductive)

The amount of overstaffing or understaffing expressed as a percentage of the actual required staff with staff adjustments.

The Summary value is an average weighted by the Staff Group's Actual Required With Staff Adjustments And Without Unproductive.

The calculation for this column is as follows:

Actual Positions Staffed / Staff Group' s Actual Required With Staff Adjustments And Without Unproductive * 100

In consolidated mode, the same calculation is performed using the consolidated values.

In a staff allocation configuration this column is blank if the staff group has a parent.

Default Title Name	AREQADH1
Internal Field Name	ActReqAdher1
Description	Actual Required Adherence 1 (This statistic is based on Staff group’s Actual Required With Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AREQADH2

Actual Required Adherence 2

(Based on Staff Group's Actual Required without Staff Adjustments and Unproductive)

The amount of overstaffing or understaffing expressed as a percentage of actual required staff without staff adjustments.

The Summary value is an average weighted by the Staff Group's Actual Required Without Staff Adjustments And Without Unproductive.

The calculation for this column is as follows:

Actual Positions Staffed / Staff Group' s Actual Required Without Staff Adjustments And Without Unproductive * 100

In consolidated mode, the same calculation is performed using the consolidated values.

In a staff allocation configuration this column is blank if the staff group has a parent.

Default Title Name	AREQADH2
Internal Field Name	ActReqAdher2
Description	Actual Required Adherence 2 (This statistic is based on Staff group's Actual Required Without Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AREQDEV 1

Actual Required Deviation 1

The amount by which actual staff deviates from the Staff Group's Actual Required With Staff Adjustments, calculated as:

Actual Positions Staffed - Staff Group's Actual Required With Staff Adjustments And Without Unproductive

This value is highlighted if it falls outside of the designated Actual vs. Required threshold. This Actual vs. Required threshold compares Actual Positions Staffed to the Staff Group's Actual Required With Staff Adjustments And Without Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

In staff allocation configurations this column is blank if the staff group has a parent.

Default Title Name	AREQDEV1
Internal Field Name	ActReqDev1
Description	Actual Required Deviation 1 *Calculation*: Actual Positions Staffed - Staff group's Actual Required With Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

AREQDEV 2

Actual Required Deviation 2

The amount by which actual staff deviates from the Staff Group's Actual Required Without Staff Adjustments, calculated as:

Actual Positions Staffed - Staff Group's Actual Required Without Staff Adjustments And Unproductive

This value is highlighted if it falls outside of the designated Actual vs. Required threshold. This Actual vs. Required threshold compares Actual Positions Staffed to the Staff Group's Actual Required Without Staff Adjustments And Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

In staff allocation configuration this column is blank for parent forecast groups.

Default Title Name	AREQDEV2
Internal Field Name	ActReqDev2
Description	Actual Required Deviation 2 *Calculation*: Actual Positions Staffed - Staff group's Actual Required Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

MAXOCC

Maximum Agent Occupancy

The maximum percentage of scheduled work time that you want agents to be actually handling contacts (including wrap-up work), as opposed to waiting for contacts. This constraint is set in the staffing scenario.

In consolidated mode, the forecast group values are weighted by each Staff Group's Revised Required.

Default Title Name	MAXOCC
Internal Field Name	MaxOccPct
Description	Maximum Agent Occupancy
Min. Decimals	0
Max. Decimals	2
Min. Value	0.01
Max. Value	100.00
Table Name	IDP_STF_GRP_DET
Field Name	MAX_OCC_PCT

OABSREQADH

Absolute Original Required Adherence

(Based on Staff Group's Original Required without Staff Adjustments and without Unproductive)

This percentage measures adherence to the Original Required staff without regard to whether non-adherence is positive or negative. For example, if 10 agents are required and 12 are actually working, the absolute adherence is 80%. Likewise, if 10 agents are required and only 8 are actually working, the absolute adherence is also 80%. This percentage is thus based on the absolute value of the Original Required Deviation.

It is calculated as:

(Staff Group's Original Required - | Original Required Deviation |) / Staff Group's Original Required * 100

The Summary value is an average weighted by the Staff Group's Original Required Without Staff Adjustments And Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	OABSREQADH
Internal Field Name	AbsOrgReqAdher
Description	Absolute Original Required Adherence (This statistic is based on Staff group’s Original Required Without Adjustments Without Unproductive.)

Min. Decimals	0
Max. Decimals	1
Min. Value	-9999.9
Max. Value	9999.9
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OREQADH

Original Required Adherence

(Based on Staff Group's Original Required without Staff Adjustments and Unproductive)

The amount of overstaffing or understaffing expressed as a percentage of the Original Required Staff Without Staff Adjustments.

The calculation for this column is as follows:

Actual Positions Staffed / Staff Group's Original Required * 100

The Summary value is an average weighted by the Staff Group's Original Required Without Staff Adjustments And Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	OREQADH
Internal Field Name	OrgReqAdher
Description	Original Required Adherence (This statistic is based on Staff group’s Original Required Without Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

OREQDEV

Original Required Deviation

The amount by which the actual staff deviates from the Original Required Without Staff Adjustments, calculated as:

Actual Positions Staffed - Staff Group's Original Required Without Staff Adjustments And Unproductive

This value is highlighted if it falls outside of the designated Actual vs. Required threshold. This Actual vs. Required threshold compares Actual Positions Staffed to the Staff Group's Original Required Without Staff Adjustments And Unproductive.

In consolidated mode, the same calculation is performed using consolidated actual and required values.

Default Title Name	OREQDEV
Internal Field Name	OrgReqDev
Description	Original Required Deviation *Calculation*: Actual Positions Staffed – Staff group's Original Required Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

Other Shrinkage Percent

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	OthShrPct
Description	Other Shrinkage Percent
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	99.99
Table Name	IDP_STF_GRP_DET
Field Name	OTH_SHR_PCT

RABSREQADH 1

Absolute Revised Required Adherence 1

(Based on Staff Group's Revised Required with Staff Adjustments and without Unproductive)

This percentage measures adherence to the Revised Required staff without regard to whether the non-adherence is positive or negative. For example, if 10 agents are required and 12 are actually working, the absolute adherence is 80%. Likewise, if 10 agents are required and only 8 are actually working, the absolute adherence is also 80%. This percentage is thus based on the absolute value of the Revised Required Deviation 1.

It is calculated as:

(Staff Group's Revised Required With Staff Adjustments and Without Unproductive - | Revised Required Deviation 1 |) / Staff Group's Revised Required With Staff Adjustments And Without Unproductive * 100

The Summary value is an average weighted by the Staff Group's Revised Required With Staff Adjustments And Without Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	RABSREQADH1
Internal Field Name	AbsRvsReqAdher1
Description	Absolute Revised Required Adherence 1 (This statistic is based on Staff group's Revised Required With Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	1
Min. Value	-9999.9
Max. Value	9999.9
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RABSREQADH 2

Absolute Revised Required Adherence 2

(Based on Staff Group's Revised Required without Staff Adjustments and Unproductive)

This percentage measures adherence to the Revised Required staff without regard to whether the non-adherence is positive or negative. For example, if 10 agents are required and 12 are actually working, the absolute adherence is 80%. Likewise, if 10 agents are required and only 8 are actually working, the absolute adherence is also 80%. This percentage is thus based on the absolute value of the Revised Required Deviation 2.

It is calculated as:

(Staff Group's Revised Required Without Staff Adjustments and Unproductive - | Revised Required Deviation 2 |) / Staff Group's Revised Required Without Staff Adjustments And Unproductive * 100

The Summary value is an average weighted by the Staff Group's Revised Required Without Staff Adjustments And Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	RABSREQADH2

Internal Field Name	AbsRvsReqAdher2
Description	Absolute Revised Required Adherence 2 (This statistic is based on Staff group's Revised Required Without Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	1
Min. Value	-9999.9
Max. Value	9999.9
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RREQADH 1

Revised Required Adherence 1

(Based on Staff Group's Revised Required with Staff Adjustments and without Unproductive)

The amount of overstaffing or understaffing expressed as a percentage of the Revised Required Staff With Staff Adjustments.

The calculation for this column is as follows:

Actual Positions Staffed / Staff Group's Revised Required With Staff Adjustments And Without Unproductive * 100

The Summary value is an average weighted by the Staff Group's Revised Required With Staff Adjustments And Without Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	RREQADH1
Internal Field Name	RvsReqAdher1
Description	Revised Required Adherence 1 (This statistic is based on Staff group's Revised Required With Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RREQADH 2

Revised Required Adherence 2

(Based on Staff Group's Revised Required without Staff Adjustments and Unproductive)

The amount of overstaffing or understaffing expressed as a percentage of the revised, unadjusted required staff.

The Summary value is an average weighted by the Staff Group's Revised Required Without Staff Adjustments And Unproductive.

The calculation for this column is as follows:

Actual Positions Staffed / Staff Group's Revised Required Without Staff Adjustments And Unproductive * 100

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	RREQADH2
Internal Field Name	RvsReqAdher2
Description	Revised Required Adherence 2 (This statistic is based on Staff group's Revised Required Without Adjustments Without Unproductive.)
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RREQDEV 1

Revised Required Deviation 1

The amount by which actual staff deviates from the revised and adjusted required staff, calculated as:

Actual Positions Staffed - Staff Group's Revised Required With Staff Adjustments And Without Unproductive

This value is highlighted if it falls outside of the designated Actual vs. Required threshold. This Actual vs. Required threshold compares Actual Positions Staffed to the Staff Group's Revised Required With Staff Adjustments And Without Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	RREQDEV1
Internal Field Name	RvsReqDev1

Description	Revised Required Deviation 1 *Calculation*: Actual Positions Staffed – Staff group's Revised Required With Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RREQDEV 2

Revised Required Deviation 2

The amount by which actual staff deviates from the revised, unadjusted required staff, calculated as:

Actual Positions Staffed - Staff Group's Revised Required Without Staff Adjustments And Unproductive

This value is highlighted if it falls outside of the designated Actual vs. Required threshold. This Actual vs. Required threshold compares Actual Positions Staffed to the Staff Group's Revised Required Without Staff Adjustments And Unproductive.

In consolidated mode, the same calculation is performed using the actual and required consolidated values.

Default Title Name	RREQDEV2
Internal Field Name	RvsReqDev2
Description	Revised Required Deviation 2 *Calculation*: Actual Positions Staffed – Staff group's Revised Required Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

RREQTS

Revised Required Update Timestamp

The date and time at which the Staff Group's Revised Required field was first written or last changed.

Default Title Name	RREQTS
Internal Field Name	RvsReqStfUpdTS
Description	Revised Required Update Timestamp
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	IDP_FOR_GRP_DET
Field Name	RVS_REQ_STF_UPD_TS

SCHADH 1

Scheduled Adherence 1

(Based on Staff Group's Original Scheduled)

The amount of actual overstaffing or understaffing expressed as a percentage of the original scheduled staff.

The calculation for this column is as follows:

Actual Positions Staffed / Original Scheduled * 100

The Summary value is an average weighted by the Staff Group's Original Scheduled.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	SCHADH1
Internal Field Name	SchAdher1
Description	Scheduled Adherence 1 (This statistic is based on Staff group’s Original Scheduled.)
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999.99
Table Name	This field is calculated at runtime.

Field Name	This field is calculated at runtime.

SCHADH 2

Scheduled Adherence 2

(Based on Staff Group's Revised Scheduled with Staff Adjustments)

The amount of actual overstaffing or understaffing expressed as a percentage of the revised scheduled staff with staff adjustments.

The calculation for this column is as follows:

Actual Positions Staffed / Staff Group's Revised Scheduled With Staff Adjustments * 100

The Summary value is an average weighted by the Staff Group's Revised Scheduled With Staff Adjustments.

In consolidated mode, the same calculation is performed using the consolidated values.

SCHADH 3

Scheduled Adherence 3

(Based on Staff Group's Revised Scheduled without Adjustments)

The amount of actual overstaffing or understaffing expressed as a percentage of the revised scheduled staff without staff adjustments.

The calculation for this column is as follows:

Actual Positions Staffed / Staff Group's Revised Scheduled Without Staff Adjustments * 100

The Summary value is an average weighted by the Staff Group's Revised Scheduled Without Staff Adjustments.

In consolidated mode, the same calculation is performed using the consolidated values.

Default Title Name	SCHADH3
Internal Field Name	SchAdher3
Description	Scheduled Adherence 3 (This statistic is based on Staff group’s Revised Scheduled Without Adjustments.)
Min. Decimals	0
Max. Decimals	2
Min. Value	0.00
Max. Value	999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SCHDELTA

Schedule Tally Delta

The change to the schedule tally from unsaved schedule changes made by the current user in Official Schedule Editor or Official Segment Worksheet. When the Live IDP Updates option is selected, values in other IDP columns temporarily show the effects of these proposed schedule changes. The Schedule Tally Delta value returns to zero when the user saves the changes, cancels the unsaved changes, or closes the window.

Because the Summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

For an Allocate configuration in which staffing requirements are allocated, the values for the child staff groups are updated. But the value for the parent staff group is not aggregated from the updated child staff groups.

Default Title Name	SCH-DELTA
Internal Field Name	StfSchDelCal
Description	Staff Schedule Delta
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	99,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SCHDEV 1

Scheduled Deviation 1

The amount by which actual staff deviates from the original scheduled staff, calculated as:

Actual Positions Staffed - Staff Group's Original Scheduled

This value is highlighted if it falls outside of the designated Actual vs. Scheduled threshold. This Actual vs. Scheduled threshold compares Actual Positions Staffed to the Staff Group's Original Scheduled for this column.

In consolidated mode, the same calculation is performed using the consolidated actual and scheduled values.

Default Title Name	SCHDEV1
Internal Field Name	SchDev1

Description	Scheduled Deviation 1 *Calculation*: Actual Positions Staffed – Staff group's Original Scheduled
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SCHDEV 2

Scheduled Deviation 2

The amount by which actual staff deviates from the revised scheduled staff with staff adjustments, calculated as:

Actual Positions Staffed - Staff Group's Revised Scheduled With Staff Adjustments

This value is highlighted if it falls outside of the designated Actual vs. Scheduled threshold. This Actual vs. Scheduled threshold compares Actual Positions Staffed to the Staff Group's Revised Scheduled With Staff Adjustments in this column.

In consolidated mode, the same calculation is performed using the consolidated actual and scheduled values.

Default Title Name	SCHDEV2
Internal Field Name	SchDev2
Description	Scheduled Deviation 2 *Calculation*: Actual Positions Staffed – Staff group's Revised Scheduled With Adjustments
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SCHDEV 3

Scheduled Deviation 3

The amount by which actual staff deviates from revised scheduled staff without staff adjustments, calculated as:

Actual Positions Staffed - Staff Group's Revised Scheduled Without Staff Adjustments

This value is highlighted if it falls outside of the designated Actual vs. Scheduled threshold. This Actual vs. Scheduled threshold compares Actual Positions Staffed to the Staff Group's Revised Scheduled Without Staff Adjustments.

In consolidated mode, the same calculation is performed using the consolidated actual and scheduled values.

Default Title Name	SCHDEV3
Internal Field Name	SchDev3
Description	Scheduled Deviation 3 *Calculation*: Actual Positions Staffed – Staff group's Revised Scheduled Without Adjustments
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGANET

Staff Group's Actual Net Staff 2

The number of scheduled staff over or under the actual number of required staff. This column is calculated as follows:

Staff Group's Revised Scheduled Without Staff Adjustments - Staff Group's Actual Required Without Staff Adjustments And With Unproductive

There is no Summary value for this column.

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistic compares the Staff Group's Revised Scheduled Without Staff Adjustments to the Staff Group's Actual Required Without Staff Adjustments And With Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

In staff allocation configurations, this column is blank for child staff groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGANET
Internal Field Name	StfActAvlStf2
Description	Staff group's Actual Available Staff 2 *Calculation*: Staff group's Revised Scheduled Without Adjustments – Staff group's Actual Required Without Adjustments With Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGANET J

Staff Group's Actual Net Staff 1

The number of scheduled staff, including staff adjustments, over or under the actual number of required staff. This column is calculated as follows:

Staff Group's Revised Scheduled With Staff Adjustments - Staff Group's Actual Required With Staff Adjustments And Unproductive

There is no Summary value for this column.

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistic compares the Staff Group's Revised Scheduled With Staff Adjustments to the Staff Group's Actual Required With Staff Adjustments And Unproductive included.

In consolidated mode, the same calculation is performed using the consolidated values.

In staff allocation configurations this column is blank if the staff group has a parent.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGANET J
Internal Field Name	StfActAvlStf1
Description	Staff group's Actual Available Staff 1 *Calculation*: Staff group's Revised Scheduled With Adjustments – Staff group's Actual Required With Adjustments With Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99

Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGAREQ

Staff Group's Actual Required without Staff Adjustments and Unproductive

The number of staff required to handle the actual volume of contacts. In a single-skill configuration, this value is copied from the Forecast Group's Actual Required without Unproductive.

In a multiskill configuration, this value is calculated by the skill-based calculation process using: Actual Number of Contacts Offered, Actual AHT, and the Staff Group's Revised Scheduled Without Staff Adjustments.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

In staff allocation configurations, this column is blank for child staff groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGAREQ
Internal Field Name	StfActReqStfWoUnp
Description	Staff group’s Actual Required Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGAREQ J

Staff Group's Actual Required with Staff Adjustments and without Unproductive

The number of staff required to handle the actual volume of contacts. In a single-skill configuration, this value is copied from the Staff Group's Actual Required Without Staff Adjustments And Unproductive. Adjustments in a single-skill configuration are applied to the scheduled statistic.

In a multiskill configuration, this value is calculated by the skill-based calculation process using: Actual Number of Contacts Offered, Actual AHT, and the Staff Group's Revised Scheduled With Staff Adjustments. Adjustments in a multiskill configuration are applied through the schedule tally.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGAREQ J
Internal Field Name	StfActReqStfWoUnpWAdj
Description	Staff group’s Actual Required With Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGAREQ JU

Staff Group's Actual Required with Staff Adjustments and Unproductive

The number of staff required to handle the actual volume of contacts. The required number of staff is inflated in this statistic, because unproductive time is included in the calculation. Staff adjustments are included as well. This column is calculated as follows:

Staff Group's Actual Required With Staff Adjustments And Without Unproductive / (1 - the Unproductive Shrinkage Percentage/100).

In a multiskill staff allocation configuration, this column is blank if the staff group has a parent.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGAREQ JU
Internal Field Name	StfActReqStfWUnpWAdj
Description	Staff group’s Actual Required With Adjustments With Unproductive
Min. Decimals	0

Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGAREQ U

Staff Group's Actual Required without Staff Adjustments and with Unproductive

The number of staff required to handle the actual volume of contacts. The value is inflated to account for unproductive time. This column is calculated as follows:

Staff Group's Actual Required Without Staff Adjustments And Unproductive / (1 - the Unproductive Shrinkage Percentage/100).

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

In staff allocation configurations, this column is blank for child staff groups.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGAREQ U
Internal Field Name	StfActReqStfWUnp
Description	Staff group’s Actual Required Without Adjustments With Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGLOCK

Staff Group's ACD Data Lock Indicator

This column contains a check box to indicate whether or not the staff group's ACD statistics are locked for that period. Any data that is locked can be manually edited, but cannot be refreshed (through the ACD

interface or through the aggregation of data from child staff groups). When you edit ACD data such as APS, that period is automatically locked and a check mark is displayed.

Default Title Name	SGLOCK
Internal Field Name	StfAutoLockoutFlag
Description	Staff group's ACD data lock indicator
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	IDP_STF_GRP_DET
Field Name	AUTO_LOCKOUT_FLAG

SGNOTE

Staff Group Note

This column displays the user-supplied note for the staff group and time period.

Default Title Name	SGNOTE
Internal Field Name	StfNote
Description	Staff group note
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	ITD_STF_CALL_NOTE
Field Name	NOTE

SGONET

Staff Group's Original Net Staff

The original number of scheduled staff over or under the number originally required. This column is calculated as follows:

Staff Group's Original Scheduled - Staff Group's Original Required Without Staff Adjustments And With Unproductive

There is no Summary value for this column.

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistic compares the Staff Group's Original Scheduled to the Staff Group's Original Required Without Staff Adjustments And With Unproductive.

In consolidated mode, the same calculation as above is performed using the consolidated values.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGONET
Internal Field Name	StfOrgAvlStf
Description	Staff group's Original Available Staff *Calculation*: Staff group's Original Scheduled – Staff group's Original Required Without Adjustments With Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGOREQ

Staff Group's Original Required without Staff Adjustments and Unproductive

The number of staff required to handle the original forecast workload without staff adjustments or unproductive time. In a single-skill configuration, this is either the same value as the Forecast Group's Original Required Without Unproductive or a staffing override.

In a multiskill configuration, this is either a staffing override value or a calculation made by the skill based calculation process using Original Forecast Volume, Original Forecast AHT, and the Staff Group's Original Scheduled staff.

In all configurations, because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In an Allocate staff allocation configuration, the values for the child staff groups are filled through allocation.

In consolidated mode, this value is the sum of the individual values for each period.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGOREQ
Internal Field Name	OrgReqStf
Description	Staff group's Original Required Without Adjustments Without Unproductive

Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGOREQ U

Staff Group's Original Required without Staff Adjustments and with Unproductive

The number of staff required to handle the original forecast workload without staff adjustments. The required number of staff is inflated in this statistic, because unproductive time is included in the calculation. Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

This value is calculated as:

Staff Group's Original Required Without Staff Adjustments / (1 - the Unproductive Shrinkage Percentage/100).

In consolidated mode, this value is the sum of the individual values for each period.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGOREQ U
Internal Field Name	StfOrgReqStfWUnp
Description	Staff group’s Original Required Without Adjustments With Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGOSCH

Staff Group's Original Scheduled

The number of agents scheduled to work based on the schedule tally at the time intra-day performance forecast was created.

In an Allocate staff allocation configuration, the value for a parent group is aggregated from the child staff group.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	SGOSCH
Internal Field Name	StfOrgSchStf
Description	Staff group's Original Scheduled
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	IDP_STF_GRP_DET
Field Name	ORG_SCH_STF

SGRNET

Staff Group's Revised Net Staff 2

The number of scheduled staff over or under the Revised Required staff. The required number of staff is inflated in this statistic, because unproductive time is included in the calculation. This column is calculated as follows:

Staff Group's Revised Scheduled Without Staff Adjustments - Staff Group's Revised Required Without Staff Adjustments And With Unproductive

There is no Summary value for this column.

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistics compares the Staff Group's Revised Scheduled Without Staff Adjustments to the Staff Group's Revised Required Without Staff Adjustments And With Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGRNET
Internal Field Name	StfRvsAvlStf2
Description	Staff group's Revised Available Staff 2 *Calculation*: Staff group's Revised Scheduled Without Adjustments – Staff group's Revised Required Without Adjustments With Unproductive
Min. Decimals	0
Max. Decimals	2

Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGRNET J

Staff Group's Revised Net Staff 1

The number of scheduled staff, with staff adjustments, over or under the Revised Required staff. The required number of staff is inflated in this statistic, because unproductive time is included in the calculation. This column is calculated as follows:

Staff Group's Revised Scheduled With Staff Adjustments - Staff Group's Revised Required With Staff Adjustments And Unproductive

There is no Summary value for this column.

This value is highlighted if it falls outside of the designated Required vs. Scheduled threshold, which for this statistics compares the Staff Group's Revised Scheduled With Staff Adjustments to the Staff Group's Revised Required With Staff Adjustments And Unproductive.

In consolidated mode, the same calculation is performed using the consolidated values.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGRNET J
Internal Field Name	StfRvsAvlStf1
Description	Staff group’s Revised Available Staff 1 *Calculation*: Staff group’s Revised Scheduled With Adjustments – Staff group’s Revised Required With Adjustments With Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	-99,999,999.99
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGRREQ

Staff Group's Revised Required without Staff Adjustments and Unproductive

The number of staff required to handle the forecast workload based on a revised forecast.

In a single-skill configuration, a staffing override or the Forecast Group's Revised Required Without Unproductive is used for this statistic.

In a multiskill configuration, a staffing override can be used or the value can be calculated by the skill-based calculation process using: Revised Forecast Volume, Revised Forecast AHT, and the Staff Group's Revised Scheduled Without Staff Adjustments.

For child staff groups in an Allocate staff allocation configuration, this value is filled through allocation. Allocated values are not recalculated automatically when saved.

In all configurations, because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGRREQ
Internal Field Name	RvsReqStf
Description	Staff group’s Revised Required Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGRREQ J

Staff Group’s Revised Required with Staff Adjustments and without Unproductive

The number of staff required to handle the forecast workload based on a revised forecast with staff adjustments.

In a single-skill configuration, this column is the same as Staff Group's Revised Required Without Staff Adjustments And Unproductive. (Adjustments in a single-skill configuration are applied only to the scheduled staff values.) In a multiskill configuration, this value can come from a staffing override, or it can be calculated by the skill-based calculation process using Revised Forecast Volume, Revised Forecast

AHT, and Staff Group's Revised Scheduled With Staff Adjustments. (Adjustments in a multiskill configuration are applied indirectly to required staff values, by means of the adjusted schedule tally.)

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In a multiskill staff allocation configuration if the staff group has a parent, the single skill calculation is used.

In consolidated mode, this value is the sum of the individual values for each period.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGRREQ J
Internal Field Name	RvsReqStfWAdj
Description	Staff group’s Revised Required With Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGRREQ JU

Staff Group's Revised Required with Staff Adjustments and Unproductive

The number of staff required to handle the forecast workload based on a revised forecast with staff adjustments and unproductive time. The required number of staff is inflated in this statistic, because unproductive time is included in the calculation.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

This column is calculated as follows:

Staff Group's Revised Required With Staff Adjustments And Without Unproductive / (1- the Unproductive Shrinkage Percentage/100).

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGRREQ JU
Internal Field Name	StfRvsReqStfWUnpWAdj
Description	Staff group’s Revised Required With Adjustments With Unproductive

Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGRREQ U

Staff Group's Revised Required without Staff Adjustments and with Unproductive

The number of staff required to handle the forecast workload based on a revised forecast. The required number of staff is inflated in this statistic, because unproductive time is included in the calculation.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

This column is calculated as follows:

Staff Group's Revised Required Without Staff Adjustments / (1- the Unproductive Shrinkage Percentage/100).

In consolidated mode, this value is the sum of the individual values for each period.

This value is affected by whether an associated staff group is staff-adjustable or volume-adjustable.

Default Title Name	SGRREQ U
Internal Field Name	StfRvsReqStfWUnp
Description	Staff group’s Revised Required Without Adjustments With Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGRSCH

Staff Group's Revised Scheduled without Staff Adjustments

The number of agents scheduled to work based on the latest schedule tally.

In an Allocate staff allocation configuration, the value for a parent group is aggregated from the child staff group.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	SGRSCH
Internal Field Name	StfRvsSchStfWoAdj
Description	Staff group's Revised Scheduled Without Adjustments
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.
Field Name	This field is calculated at runtime.

SGRSCH J

Staff Group's Revised Scheduled with Staff Adjustments

The number of agents scheduled to work based on the latest schedule tally, with staff adjustments included. First, any absolute adjustments are summed and applied to the Staff Group's Revised Scheduled Without Staff Adjustments. Then any percentage adjustments are summed and applied to this absolute-adjusted schedule tally.

In an Allocate staff allocation configuration, the value for a parent group is aggregated from the child staff group. Likewise, the staff adjustments are applied to the child staff groups and then used to form the parent values.

Because the summary value is expressed as staff hours, it is equal to the sum of this column divided by two (for 30 minute periods) or by four (for 15 minute periods).

In consolidated mode, this value is the sum of the individual values for each period.

Default Title Name	SGRSCH J
Internal Field Name	StfRvsSchStfWAdj
Description	Staff group's Revised Scheduled With Adjustments
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	This field is calculated at runtime.

Field Name	This field is calculated at runtime.

SGSTATUS

Staff Group's ACD Data Status Indicator

This column displays the status of the ACD data for the staff group and time period. There are four possible values:

- Complete: All ACD data has been received for a given period for a particular forecast or staff group.
- Incomplete: Indicates that the data for one or more data groups of the forecast or staff group is missing for that period.
- Unavailable: Indicates that no data has been received for the forecast or staff group for that period.
- Manual edit: The user manually changed the data.

Default Title Name	SGSTATUS
Internal Field Name	StfStatus
Description	Staff group’s ACD data status indicator
Min. Decimals	NA
Max. Decimals	NA
Min. Value	NA
Max. Value	NA
Table Name	ITD_STF_CALL_STAT
Field Name	STATUS

Stored Original Required Without Adjustments Without Unproductive

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	StrOrgReqStf
Description	Staff group’s Stored Original Required Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99

Table Name	IDP_STF_GRP_DET
Field Name	ORG_REQ_STF

Stored Revised Required Without Adjustments Without Unproductive

This column is not displayed in the software. It is used for internal calculations only.

Default Title Name	This field is not displayed.
Internal Field Name	StrRvsReqStf
Description	Staff group's Stored Revised Required Without Adjustments Without Unproductive
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	999,999,999.99
Table Name	IDP_STF_GRP_DET
Field Name	RVS_REQ_STF

UNP

Unproductive Shrinkage Percentage

The percentage of time that agents are scheduled to be handling work with standard efficiency, but are actually not. To compensate for the fact that agent schedules do not include unproductive time, the unproductive shrinkage is used to inflate the required staff values in some statistics. The unproductive shrinkage percentage comes from the shrinkage set that you have selected in the staffing scenario.

In consolidated mode, this is an average of the values for individual staff groups, weighted by each Staff Group's Revised Required.

Default Title Name	UNP
Internal Field Name	UnpShrPct
Description	Unproductive Shrinkage Percent
Min. Decimals	0
Max. Decimals	2
Min. Value	0
Max. Value	99.99

Table Name	IDP_STF_GRP_DET
Field Name	UNP_SHR_PCT

Appendix A. Security Auditing

If enabled, all changes made to the security configuration in Aspect Workforce will be logged in an audit trail making it possible to identify when security entities were modified and by whom.

Security Tables Audited

Table A-1 lists the security tables that are audited.

Table A-1 Audited Security Tables

Function	Parent Table	Child Tables
User Account	TCS_LOGIN	N/A
Security Profile (System and Instance Lock Permissions)	SECUR_PROF	GRNT_EXTRA_FIELD GRNT_FILTER_PROF GRNT_FOR_GRP GRNT_GRP_ALW_ACCT GRNT_PERS_ACCT GRNT_RTG_SET GRNT_SCH_PROF GRNT_SEAT_FILTER_PROF GRNT_SECUR_PROF GRNT_SEG_CODE GRNT_STF_GRP GRNT_SYS_LOCK
Date Window Set	DATE_WIN_SET	DATE_WIN
Employee Group Filter Profile	FILTER_PROF	EMP_GRP_FILTER
Seat Group Filter Profile	SEAT_FILTER_PROF	SEAT_GRP_FILTER

Audit Trail Tables

The audit trail tables have the name of the table being audited prefixed with “AUD_”. For example, the audit trail table for TCS_LOGIN is named AUD_TCS_LOGIN.

Table A-2 shows the format of an audit trail table.

Table A-2 Audit Trail Table Format

Table	AUD_<table>					
Privileges	None					
Column	**Type**	**Key**	**Delete**	**Update**	**Null**	**Constraints**
AUD_TS	Timestamp	PK		N	N	
AUD_SEQ_NUM	Sequence	PK		N	N	
AUD_ACTN	NVARCHAR(1)			N	N	I, U, D
AUD_TCS_LOGIN_SK	SK		DR	N	N	
...						

All of the columns from the audited table will be present in the audit trail table. The AUD_ACTN column indicates the action performed (I=Insert, U=Update, D=Delete)

Security Violation Tables

If enabled, rejected security attempts are logged any time a user attempts to perform an action in Aspect Workforce and is denied due to insufficient security privileges. The violations are logged in the AUD_SYS_LOCK_VIOL table for violations against system locks and in the AUD_INST_RESTR_VIOL table for violations against instance locks.

Table A-3 and Table A-4 document the two tables used to log security violations.

Table A-3 System Lock Violations Table

Table	AUD_SYS_LOCK_VIOL					
Privileges	None					
Column	**Type**	**Key**	**Delete**	**Update**	**Null**	**Constraints**
AUD_TS	Timestamp	PK		N	N	
SEQ_NUM	Sequence	PK		N	N	
AUD_TCS_LOGIN_SK	SK	FK	DR	N	N	
SYS_LOCK_ID	Integer	FK	DR	N	N	

NOM_DATE	Nominal date			N	Y	

Table A-4 Instance Lock Violations Table

Table	AUD_INST_RESTR_VIOL					
Privileges	None					
Column	**Type**	**Key**	**Delete**	**Update**	**Null**	**Constraints**
AUD_TS	Timestamp	PK		N	N	
SEQ_NUM	Sequence	PK		N	N	
AUD_TCS_LOGIN_SK	SK	PK	DR	N	N	
INST_REST_ID	SK	FK	DR	N	N	
NOM_DATE	Nominal date			N	Y	
SK	SK			N	N	

Appendix B. Glossary of Aspect Workforce Terms

This section defines Aspect Workforce terms that are relevant to this SDK.

abandoned contact

A contact received by the ACD but disconnected before being answered by an agent, usually because the customer loses patience.

Access Control modules

A group of administrative modules with which you identify your Aspect Workforce users, specifying which parts of the software they can use and which employee groups they can access.

accounting period

A span of time (such as a year or month) for which your organization applies credits to personal accounts and sets limits on the number of hours employees can carry over to the next period.

Accounts Manager modules

A group of modules that you use to set up personal accounts and group allowance accounts.

ACD (Automatic Contact Distributor)

A device that receives incoming contacts and either routes them to available agents or handles them in some other way (such as placing them in a queue). To Aspect Workforce, an ACD is a source of historical data about actual contacts and agent performance.

Actual Data

A module that lets you view and revise the historical contact and agent data that Aspect Workforce has received from an ACD.

ACD instance

A setup record that describes a source of ACD data to be processed by Aspect Workforce.

ACD interface

The hardware and software links that connect your ACD's reporting system to Aspect Workforce and thus enable Aspect Workforce to capture historical data.

ACD perspective

A special perspective used to keep track of when employees are supposed to be logged in to an ACD.

active employee

An employee who is currently part of your working staff and can thus have official segments. See inactive employee.

adherence

The degree to which the behavior of employees—particularly their arrivals and departures—agrees with their official schedules.

administrator

A type of Aspect Workforce user who has full access to all modules and all data, including the ability to define permissions and create and administer other users.

agent

Someone who handles contacts routed through an ACD.

Agent Productivity

A feature of the Aspect Workforce™ Perform enhancement package that analyzes ACD statistics and gives you comprehensive reports on the activities and availability of individual agents and groups of agents.

agent-initiated

Call work initiated by an agent's dialing a number. Compare system/customer-initiated.

AHT (average handle time)

The period of time an agent is occupied with an incoming contact, measured from the moment the agent answers a contact until the moment the agent has finished the wrap-up work for that contact and is considered available.

AHT distribution

A set of quarter- or half-hourly factors, for each day of the week, that defines the typical distribution of average handle times throughout the day. Each factor measures how far AHT in that period deviates from the AHT for that day as a whole.

Allocate

An enhancement package of Aspect Workforce™ Allocate. Allocate is a specially configured version of Aspect Workforce, designed for networked contact centers, that allows for both central and local forecasting, scheduling, and tracking. Depending on the methods by which contacts are distributed over the network, Aspect Workforce uses one or more types of allocation to accomplish these tasks.

allocation

The division of work, staffing requirements, or schedules among several locations, by percentage and by time of day. These locations can be independently operating contact centers or management teams responsible for their own scheduling or adherence.

allocation set

A list of the percentages by which to allocate contact volumes, workload, or required staff among forecast groups or staff groups. You create an allocation set for each parent group and either calculate or manually enter the allocation percentages, by time of day, for each of its child groups.

Allocation Set modules

The Forecasting modules you use to create and update allocation sets.

allowance account

An account that tracks how many hours are available to employees for vacation, personal time, or other activities.

anchor segment

In a segment window-rule, the fixed segment (usually a container) in relation to which a break (or similar segment) is to be scheduled.

APParser

A Windows system service that is identical to TCSParser except that it processes ACD data for Agent Productivity, an application in Perform. Aspect Workforce distinguishes between the two because Agent Productivity requires different ACD reports than Aspect Workforce.

application server

A computer used for Aspect Workforce operations that require centralized processing or enforcement of complex business rules.

assignment

The association of individual employees with specific work schedules. Also, the association of individual employees with employee groups.

attempt

The dialing of a number, whose outcome includes, but is not limited to, the following: answering machine answered, line is busy, connected to right party, connected to wrong party, abandoned by dialed party.

automatic assignment

A process that automatically assigns employees to trial schedules according to their schedule preferences and a user-defined employee order.

automatic historical pattern update

A process that uses selected actual historical data to update a forecast group's monthly or intra-day historical patterns without user intervention.

AutoRun

A utility that executes Aspect Workforce processes (or other applications) on a scheduled basis, eliminating the need for users to execute those processes interactively.

AutoRun Manager

A module that you use to set up and manage jobs to be executed by AutoRun.

AutoRun queue

A list of scheduled processes that AutoRun reads to determine which ones it needs to execute, and when.

AutoRun workstation

The computer on which the AutoRun software is installed (as a system service) and on which all AutoRun processes are executed. This is normally the application server.

available hours

For a given staff group, the time periods for which Aspect Workforce will calculate staffing requirements. It is assumed that these are the only periods in which employees with the required skill association will be available for work. In addition to the normal hours (which recur week after week), you can define special hours that override the normal settings during specific periods on specific dates.

average daily volume

A historical pattern that represents what a forecast group's daily contact volume would be if there were no daily or monthly fluctuation. This is the value to which all other forecast-group patterns are applied.

average delay371

The average amount of time between the moment a contact is received by the ACD and the moment it is answered by an agent. Average delay typically includes time for an initial recorded announcement plus time spent waiting in queue. This statistic can be used as the basis for calculating required staff.

average speed of answer (ASA)

See average delay.

back-office work

Tasks that are normally handled outside of the contact center and have long service delays, such as claims or order processing. Unlike contacts, back-office tasks wait in queue without abandoning, and their non-random arrival rate simplifies staffing calculations.

balance over/under

A stopping rule that tells the scheduling process to create only enough schedules to achieve a net balance of understaffing and overstaffing for the entire scheduling period.

Balance Viewer

An advanced tracking module that lets you view current personal account balances, as well as balances for group allowances and intra-day staffing changes.

band

Two or more related columns, in a grid or a report, that are grouped together under a top-level heading.

base staff

A group of pre-existing schedules that you can use as a starting point in generating schedules. Any new schedules created are in addition to the base schedules.

bodies in chairs

A method or unit of measure for expressing staffing requirements, in which shrinkage is not taken into account. The required number of staff thus represents agents who are actually handling (or waiting for) contacts, as opposed to agents who are engaged in other activities or who are not working as productively as expected.

break optimization

The automatic adjustment of break start times (or other segment start times) in schedules so as to more closely match scheduled staff to required staff in each period of the day.

budget forecast

A special type of forecast that provides information useful for planning your contact center's budget and hiring needs.

buffer time

A period immediately before or after a seat reservation that allows the employee to prepare for, or wrap up, the work session. Pre- and post-buffer times are configured for each seating state.

campaign

A dialing plan used to contact a contact list. Consists of the size of the contact list for a specific day and the number of passes through the list.

campaign set

A cyclic pattern that adjusts the daily volume forecasts for a range of dates by distributing additional contacts among them.

character

Any symbol that can be typed from the keyboard, usually a letter, number, space, or punctuation mark.

Checker rule

A specific test you create for the Checker process to use in automatically approving or denying segment requests. Each rule specifies not only the conditions that must be met for approval, but also the selection criteria that determine which requests will be checked against that rule.

Checker

An automatic process that tests each segment request against certain account balances and user-defined rules and, on that basis, approves the request, denies it, or places it on a waiting list. Checker runs as a system service on an application server.

child group

In a tree or other hierarchical relationship, a group that is the immediate subordinate of another group (parent). This can be any of the following:

- An employee group that represents a subdivision of another group; for example, one of two work teams that are part of a larger organization.
- In an Allocate configuration, a forecast group whose forecast volume, AHT, and workload are allocated from another.
- In an Allocate configuration, a staff group whose staffing requirements, adherence requirements, or trial schedules are allocated from another.

client workstation

A computer that you use to run the Aspect Workforce software interactively, as opposed to one of the server computers.

client/server

An approach to data processing in which the functions of data storage and retrieval are handled by a different program (server) from that used for computation and user interaction (client). Typically one server handles requests from multiple clients, all running on different machines. The architecture of Aspect Workforce, however, uses a three-tier approach.

completion percentage

Number of connections divided by the number of attempts. Typically expressed by day of week and period of day.

compliance

An employee's adherence to scheduled start and stop times over the course of a day. Employees with perfect compliance don't arrive late or depart early. They either arrive during or before scheduled start times or depart during or after scheduled stops. In no case can an employee achieve more than 100% compliance. For example, starting 5 minutes early and leaving 5 minutes late will not result in 105% compliance.

Configuration modules

A group of modules with which you create a comprehensive description of your contact center's structure, the work it handles, and the various codes used to maintain employee information and track employee time.

connection

The completion of a call. Can be of three types: right-party connections, wrong-party connections, and abandoned calls. This term is used interchangeably with "completion."

consolidation

In intra-day performance forecasting, the combining of like statistics for two or more forecast groups, or two or more staff groups, from the same routing set.

contact center

Any business operation that receives customer contacts and uses an ACD to route them to agents.

Contact Center Calculator

A module that uses Erlang formulas to quickly solve various staffing and trunking problems without reference to historical patterns.

contact distribution

A set of quarter- or half-hourly percentages, for each day of the week, that defines the typical distribution of contact volumes throughout the day. Each percentage measures how far that period's volume deviates from the average volume for that day as a whole. Also known as *volume* or *NCO* distribution.

contact list

A list of accounts (that is, telephone numbers) that are to be called as part of a campaign.

container segment

A segment that has no purpose other than to mark the beginning and end of a shift and thus serve as a "container" for other segments. It is a low-ranking segment that produces the state *Idle* in the Staff perspective and the state *Normal* in the Pay perspective.

cost-vs.-revenue optimization

A staffing basis in which the required number of staff is assumed to be that which maximizes net revenue.

cycle cut set

A cyclic pattern that adjusts the daily forecast volumes for a range of dates without affecting the fiscal forecast totals.

cyclic pattern

A historical pattern that modifies the daily forecast volumes for a range of dates in order to model multiday contact patterns associated with particular events, such as marketing promotions, billings, and catalog mailings.

daily average handle times

A historical pattern consisting of a single AHT for each day of the week.

data group

A setup record that lets you map ACD statistics to forecast groups and staff groups. A data group identifies one or more sources (report names, ACD group ID numbers, etc.) of ACD data, as well as the formula needed to correctly interpret the data.

Data group set

A setup record that associates ACD groups with ACD reports and contact, agent, or AP formula sets.

database administrator

An Aspect Workforce user—typically a system administrator—who is authorized to do setup, maintenance, and administrative tasks associated with the Aspect Workforce database.

database server

The software that maintains the Aspect Workforce database, in which is stored all Aspect Workforce data shared by Aspect Workforce users (such as employee information, forecasts, work schedules, and so on).

date window

An administrator-defined range of dates that can be used to set date-sensitive restrictions on accessing or exporting official schedule information.

day-of-week factors

A historical pattern consisting of seven factors, one for each day of the week, that defines the typical distribution of contact arrival throughout the week. Each factor measures how far contact volume on that day deviates from the average daily contact volume.

default

A value or choice assumed by a program in case you give no instructions to the contrary. For example, a program might prompt you for a date and use today's date if you do not specify another.

default employee filter

The employee filter settings that are automatically in effect whenever you open a module that is affected by the employee filter. When you change the employee filter settings for a particular module, you can save them as the new default filter.

default scenario

The scenario that you have designated to be used by default for a given forecast group or staff group in operations such as forecasting, scheduling, and intra-day performance.

default schedule preferences

One or more schedule preferences that can be used in automatic assignment or preference-based scheduling for employees who have no individual preferences in the preference set being used, or whose preferences fail to match an available schedule.

default state

The state that is automatically in effect whenever an employee has no segments that are associated with that perspective. You designate the default state (on the State dialog box) when you're defining the perspective.

default tracking date375

The tracking date that Aspect Workforce assumes automatically whenever you open a module. You can change your default tracking date using Tools > Options.

detail segment

An ordinary segment, having a specific start time and stop time, as opposed to a general segment, which has no start time or stop time of its own.

discretionary segment

A segment that removes an employee from his or her primary work duties but that can be canceled on short notice. By having a category for discretionary segments, you can easily identify them when you are looking for additional employees to cover the primary work.

display set

A user-created set of intra-day performance statistics (columns) that you can choose to display for a given forecast group, staff group, routing set, or (with conventional, single-skill routing) forecast group / staff group association.

dropout rate

The potential or tendency of customers waiting in queue to abandon, expressed as the number of abandoned contacts per hour.

economic assumptions

A set of forecasting scenario assumptions that defines cost and revenue components for use in budget forecasting and staffing requirements calculation.

employee filter

A set of instructions that tells Aspect Workforce which employee information to retrieve, or not retrieve, from your database server. This option is available in several Aspect Workforce modules.

employee group

Any division or category that you have created to organize your employee information; a point on an employee tree.

employee group class

A user-defined category that describes similar employee groups in the same employee tree. For example, a tree might consist of several Offices, each having several Teams. Using classes gives you the option to group employee-specific report detail by employee group.

employee information

A broad category of data that you maintain about each of your employees, including both administrative information and information used in schedule assignment and tracking.

Employee Information module

A module that you use to view and update an individual employee's information.

employee supergroup

A group of employee groups, typically representing employees with similar job functions or organizational status. In request management, the employee groups in a given supergroup all draw from the same group allowance account(s).

employee tree

Any of several hierarchical structures by which your employees can be organized; for example, by site, by supervisor group, or by employee category.

employee user

An type of Aspect Workforce user, normally a contact-center agent, who has an employee information record. Employee users have access to only certain software features and to data that applies only to **themselves.**

Employees modules

A set of modules you use to maintain information about your employees, including their work skills and group assignments.

Empower

An enhancement package and its associated software components: mainly Notification Server.

Encompass

An enhancement package that facilitates the exchange of workforce management information, such as number of required staff, with outsourced staff providers. Using Allocate features along with data import and export features, an Encompass client site can integrate outsourced staff with its own for forecasting, scheduling, and tracking.

entitlement

A specific amount of time credited to an employee's personal account. Entitlements are typically earned as employees accumulate service time, in which case the amount of each employee's credit depends on the length of service as measured from the employee's hire date. You can also issue entitlements for fixed amounts unrelated to length of service.

entitlements table

A table you create that specifies the amount of each entitlement an employee earns for each length of service (number of months employed).

Erlang formula

A mathematical method of making predictions about randomly arriving workload based on known information (such as average contact duration). Although traditionally used in telephone traffic engineering (to determine the required number of trunks), Erlang formulas have applications in contact center staffing as well.

Event Viewer

A module that lets you view a detailed log of the results of actions attempted by individual Aspect Workforce users and processes.

extra fields

A set of general-purpose fields in each employee record that you can use to store any information you choose. These are user-defined fields that you create and label yourself.

extra information

The portion of employee information that is stored in extra fields.

factor

A number by which another number is multiplied as part of a calculation. In Aspect Workforce, a factor is typically a constant that is applied to varying quantities (such as contact volumes or contact handle times).

field

A single unit of data (such as an employee ID) or parameter setting (such as Yes or No).

filter

A set of instructions that tells Aspect Workforce which information to retrieve, or not to retrieve, from your database. For example, Aspect Workforce can display a list of work schedules more quickly if you tell it to filter out—that is, not to retrieve—schedules for past dates. Making a change to filter settings already in effect generally results in a new query.

filter profile

A set of permissions that, when associated with a user, specifies which employee groups or combinations of employee groups that user can access.

first day of week

The day of the week that a user considers to be the first day of the work week. This option is used only for display and reporting purposes. You can set your own first day of week using Tools > Options.

fiscal calendar

A list of dates that define the reporting periods you want to use for forecasting, planning, and other applications. In addition to the standard monthly calendar, you can create any kind of fiscal calendar you need: weekly, biweekly, quarterly, semiannual, and so on.

fiscal period

One unit of time in a fiscal calendar, defined by its beginning date and (implicitly) by the beginning date of the next period in sequence.

forecast

A prediction of future contact volumes, workloads, or required staff by month or fiscal period, by date, and by intra-day period.

forecast adjustment

A manual change you can make, in advance, to the forecast number of contacts in a month. This is useful for “what if” analyses or for months that you expect to be abnormal.

forecast group

The business unit for which Aspect Workforce forecasts contact volume and workload, defined as a work type as it appears at a particular location. You create a forecast group so you can get contact volume forecasts for that location, which in turn provides a basis for forecasting staffing requirements.

Forecaster modules

The forecasting modules that you use to create forecasts (as opposed to the other forecasting modules, which you use to update assumptions).

forecasting basis

Any one of six alternative scenario assumptions used to determine how a volume forecast will be generated. The forecasting basis tells Aspect Workforce whether to use historical patterns, user-supplied volumes or adjustments, or some combination of these.

forecasting scenario

A set of objectives and assumptions used in forecasting contact volume and workload. Each forecasting scenario is associated with a specific forecast group.

form

A special dialog box that you use to add, edit, or view information stored in your Aspect Workforce database.

FTE (full-time equivalent)

A scenario assumption used in budget forecasting and scheduling that defines the number of hours per week full-time employees are normally supposed to work. As a measure of staffing level, an FTE is equivalent to a full-time position, even though the hours may actually be filled by part-time schedules.

full coverage

A stopping rule that tells the scheduling process to create enough schedules to ensure that the forecast contacts can be answered within the established service quality requirement.

general segment

A segment that, having no specific start time or stop time of its own, takes on the start times and stop times of the detail segments for that same nominal date. Typically (but not necessarily) this is a high-ranking segment that overrides the employee's other segments for the day.

generate

To create multiple items automatically—in particular, trial schedules—based on assumptions, rules, schedule templates, and so on.

group allowance

The total amount of time off, extra work time, or similar time benefits that you will allow a supergroup of employees to exercise on a given date, insofar as a particular staff group's requirement is concerned.

group assignment

A record assigning a particular employee to an employee group as of a certain date.

group scheduling

A type of preference-based scheduling that considers the requirements of scheduling groups. You can use group scheduling to assign schedule preferences and constraints to an employee group whose employees need to begin or end their schedule at about the same time, or who need to attend a meeting together.

growth rate

The historical annual change in contact volume, expressed as a percentage increase or decrease and calculated by Aspect Workforce based on actual ACD data. For example, if last year's volume was 1,100,000 contacts and the previous year's volume was 1,000,000, then the growth rate for last year was 10%.

handled contact

A contact that is answered by an agent, as opposed to being abandoned.

historical data

Information reported by the ACD about a work type's recent contact volume and average handle times and used by Aspect Workforce to derive historical patterns. This is in the form of daily contact volume totals and half- or quarter-hourly contact volumes and AHTs.

historical pattern scenario

A set of parameters used in automatically updating historical patterns for a specific forecast group. Each scenario assigns actual data, by date, to selected intra-day patterns and daily AHT sets.

historical patterns

The statistical representations of your past contact arrival and AHT patterns, as well as staff shrinkage, stored by Aspect Workforce for use in forecasting and scheduling.

Historical Patterns modules

The Forecasting modules that you use to regularly update your historical patterns. Each module applies to either forecast groups or staff groups.

holiday factor

A historical pattern associated with a specific date and multiplied by the forecast contact volume for that date in order to take into account an expected increase or decrease in volume. For example, if on a given day only half the usual number of contacts occur for that day of the week and that time of year, the holiday factor for that date would be .5.

Housekeeper modules

A group of Administration modules that you use for various maintenance processes such as archiving and deleting old data.

icon bar

The rightmost area of the Aspect Workforce main window, where you can create buttons for modules that you want to be able to open quickly.

inactive employee380

An employee who is not currently part of your working staff. Inactivating an employee prevents official segments from being created or changed for that employee. You can also set a user option to prevent inactive employees from appearing in your on-line views and reports.

internal growth rate

The historical growth rate on which forecasts are usually based, as opposed to a growth rate you have entered yourself.

intra-day distribution

A historical pattern, consisting of percentages or factors for each intra-day period, that defines the expected distribution of contact arrival or average handle time throughout each day. Each number measures how far contact volume or average handle time in that half hour or quarter hour deviates from the average half-hourly or quarter-hourly figure for that day.

intra-day performance (IDP) forecast

A continually updated forecast of contact volume, average handle time, required staff, and service quality for each period (half or quarter hour) of a specific day. This is a comprehensive analysis based on historical patterns, scenario assumptions, and current schedules.

Intra-Day Performance modules

A group of modules that display intra-day performance forecasts, providing you with tools for making intelligent staffing adjustments as actual circumstances change.

intra-day staffing balance

The allowable increase or decrease in staff for one period of the day. This is based on the staffing tolerances set by your organization to ensure that an appropriate number of staff will be on duty at all times.

Intra-Day Time Line

A module that shows you the states resulting from the segments for selected employees on a particular date.

intra-day work assignments

In an intra-day work schedule: two or more work segments that are for different staff groups. You can create intra-day work assignments automatically using the Optimize Work Assignments option of Segment Worksheet.

Journal modules

Two modules that let you enter, edit, and view a memo for any date, by forecast group or by staff group. You can thus keep records of significant events and other information that other users can access.

LAN (local area network)

A short-distance network (usually within a single office or building) that links computers and peripheral devices under some form of standard control.

legal holiday

Any holiday for which special wages are paid to employees who work on that day and for which you want holiday hours to be summed for reporting. You can designate a legal holiday when you create a holiday factor.

location

Any site, department, office, work team, or other business unit that participates in the receipt, delivery, or handling of work.

Login Manager

A module that lets you view users logged in to Aspect Workforce and (depending on your user permissions) perform related administrative tasks, such as resetting passwords.

master template

A type of schedule template that you use to assemble shift templates into a single sequence of days and set rules that apply to that entire sequence.

meeting

Two or more coincident segments, for different employees, that represent a shared activity. The segments that make up the meeting all have the same codes, times, and durations.

memo

A free-form field used to store descriptive text or comments. The information in a memo field can be of any length. As you type, text automatically wraps at the end of each line, and a scroll bar appears if the memo length exceeds the visible display space.

minimum interbreak interval

A scheduling assumption specifying the minimum amount of time that must elapse between the end of one break and the beginning of another.

module

A distinct, functional unit of the Aspect Workforce client software that you use to accomplish related tasks. For example, you use the Employee Records module to maintain information about your employees.

monthly update

A procedure you do each month to update monthly and daily historical patterns using actual daily contact volume totals for input.

multiple-schedule assignment

A method of automatic assignment in which each employee can be assigned to several one-day schedules in a given run. This method is also used in preference-based scheduling.

multi-select

To select several items in a list or grid so you can carry out the same operation for all of them at once. You can multi-select by holding down the Ctrl key as you click individual items or by holding down the Shift key and clicking items to select a continuous block. Using the Select All option is a convenient way to select all items.

multiskill configuration

An Aspect Workforce configuration option that lets you model skill-based routing in forecasting staffing requirements, scheduling, and tracking intra-day performance.

multiskill routing set

A routing set in which a forecast group is associated with more than one staff group. Normally, at least one of those staff groups is a multiskilled group and is therefore associated with more than one forecast group.

net calls

In a routing set, the number of calls in a forecast group that can be removed from or added to the number of offered calls such that the forecast group's service quality goal is met. This number is equal to the forecast calls minus the required calls.

net revenue

The potential revenue to be gained from contacts in a particular period, minus the costs associated with handling those contacts. Such costs can include wage costs, telephone costs, and lost revenue due to abandoned contacts.

net staff

The actual number of staff minus the required number of staff in a given period. Net staff that is positive represents overstaffing; net staff that is negative, understaffing.

nominal date

The date that a segment is associated with for reporting and selection purposes. For example, if you select schedule information for a particular tracking date, a segment having that same nominal date will be included even though it might actually start on the previous date.

normal distribution

A seven-day intra-day distribution (contacts or AHT) that represents the contact patterns that are typical for each day of the week. Each staff group has a default normal distribution of each type, which is used in each forecast unless you specify otherwise.

normal operating hours

The operating hours that are defined by day of the week and thus create a repeating pattern, as opposed to special operating hours, which are defined for specific calendar periods.

normalize

To change an unusual contact statistic reported by the ACD so as to reflect what would have been typical for that period of the day or that day of the week. Normalization is done (whether manually or automatically) before updating historical patterns, so that the historical patterns will not be distorted by the unusual data.

occupancy

The percentage of the scheduled work time that agents are actually handling contacts or wrap-up work, as opposed to waiting for contacts.

offered contact

A contact that is received by the ACD. Offered contacts are either handled by an agent or abandoned by the customer.

official schedule

All of the official segments associated with a given employee. An official schedule is not a thing in itself, but simply a convenient concept for identifying an employee’s official segments for a given range of dates.

Official Schedule Editor

A module that you use to add, edit, or delete official segments for a given employee—or that an employee uses to enter a segment request.

official segment

A segment for a specific employee that represents how the employee will actually be spending time (or did spend time), as opposed to a segment that's part of a trial schedule. Official segments can either be created from trial schedules or entered manually as the need arises.

Official Segment Worksheet

A module that you can use to quickly add, edit, or delete official segments and save your changes as one batch.

one-way trade

A schedule trade in which only one employee is working the other's schedule.

operating hours

For a given forecast group, the time periods in which work is allowed to arrive. It is assumed that work arriving outside of those periods is either blocked or routed somewhere else. In addition to normal operating hours, you can define special operating hours that override the normal settings during specific periods on specific dates.

outbound campaign pattern

A type of historical pattern that describes the workload for outbound connections for specific days and periods of the day. Consists of outbound parameters that summarize actual outbound data. Use outbound campaign patterns in the Outbound Campaign Patterns module to generate an outbound campaign plan

outbound campaign plan

An outbound forecast or dialing plan, including number of attempts and completions. Generated by applying input variables such as contact list size and number of passes to an outbound campaign pattern.

override set

A set of user-supplied, intra-day values—contact volumes, AHT, or required staff—used as input to forecasting or scheduling runs. An override value is used in place of a value that would have otherwise been calculated.

Overrides modules

The modules you use to create and update override sets.

parent group

In a tree or other hierarchical relationship, a group that is the immediate superior of another group (child). This can be any of the following:

- An employee group that is subdivided.
- In an Allocate configuration, a forecast group whose forecast volume, AHT, and workload are allocated to others.
- In an Allocate configuration, a staff group whose staffing requirements, adherence requirements, or trial schedules are allocated to others.

pass

In the Outbound Campaign Patterns module, the dialing of all numbers in a contact list, whether by an agent, a dialer, or both.

Pay perspective

A special perspective used to keep track of when employees are earning pay and which categories of pay they are earning.

peak position requirements

The maximum number of staff required in any half hour within a given date range.

percentage of required staff

A stopping rule that tells the scheduling process to create only enough schedules such that no more than a certain percentage of the calculated staffing requirements are covered in any one period.

performance worksheet

An interactive forecast of back-office work, and its handling, for a specified routing set. To create and update performance worksheets, use the Performance Worksheet module.

permanent closeout

A maintenance process that deletes all official segments and superstate summary hours for all employees through a specified date. You initiate this process using the Permanent Closeout module. When the data is deleted, all superstate transactions in the personal account through that date are combined and replaced by one closeout transaction.

personal account

An account that keeps a running balance of how much time is available to an individual employee for activities such as vacation, personal time off, and training.

personal accounting

A feature that lets you track the amount of time earned by individual employees in various categories (accounts) such as vacation, as well as the amount of time each employee has already used. You can use personal account balances as one of the criteria for approval in request management.

perspective

An area of interest, or point of view, for describing the effects or implications of various schedule segments. For example, taking a 15-minute break has a negative effect from the perspective of staffing levels but a neutral effect from the perspective of whether or not the employee is being paid.

pop-up menu

A menu that appears only when you summon it by right-clicking a particular object or area in a window (also known as *context* menu). Each pop-up menu provides the available commands that area most frequently needed for that object or area.

preference set

A set of schedule preferences, for any number of employees, that you can select for a given automatic assignment or preference-based schedule run. In principle, each set addresses a particular need or set of assumptions—for example, “what types of schedules employees prefer during the summer months.”

preference-based scheduling

An approach to generating trial schedules in which employee preferences are considered along with staffing requirements and each schedule is assigned to an employee when it is created.

project

A special task, such as filing or transaction processing, that requires agents to be unavailable for answering contacts.

query

A request that Aspect Workforce sends to your database server in order to retrieve a list of items having specified characteristics. A new query generally occurs when you open a list, when you refresh it or sort it, or when you change its filter settings.

queue

An ACD feature that holds incoming contacts when no agents are available to answer them. Contacts in queue are individually tracked and routed to agents as they become available, with the oldest contacts normally getting the highest priority.

reforecasting

In intra-day performance, the recalculation of estimated workload and related statistics for future periods, based on emerging patterns in the periods that have already elapsed.

refresh

To update your current view of the contents of the Aspect Workforce database, as well as any information derived or calculated from it. This occurs either automatically or as a result of choosing the View > Refresh option. Aspect Workforce refreshes your view by repeating the last query.

Regional Settings

A Windows Control Panel option with which you choose the formats to use for dates, times, and currency. Aspect Workforce follows the settings you choose.

regular user

A type of Aspect Workforce user whose access to specific software features and data is controlled by permissions (security profile and filter profile). Regular users typically include supervisors, managers, analysts, data-entry staff, and so on.

Report formula

A collection of settings that tell Aspect Workforce how to extract contact volume, staffing, and other data from ACD reports. Note that you modify formula sets only if you created them. Default sets are read-only. A report script uses the settings specified in a report formula to extract needed data from your ACD reports.

Report script

A set of instructions used by the Aspect Workforce ACD data processing engine to convert raw ACD data to a format that Aspect Workforce can use. Each script processes a specific report from a specific ACD. Report scripts are created by the Alvaria software development staff and typically are not modified by Aspect Workforce users. Report scripts are specified within an ACD instance.

Reports modules

A group of modules with which you can configure, view, print, and export reports on various Aspect Workforce data.

request bidding

A WFO feature: a project, or process, in which a number of employees enter requests for vacation, overtime, or similar schedule changes and those requests are evaluated later, in a specific order (such as seniority), as opposed to the order in which they were entered. A *request bid* is the set of data that identifies the requests to be considered, identifies the processing options (request scenario) to be used, and tracks the results of request processing.

request management

An Aspect Workforce feature that lets you enter employee requests for time off, overtime, or similar schedule changes and submit those requests to an automatic checking process, which you have tailored to your own rules and policies.

Request Manager

A module with which you can check or change the status of segment requests.

Request Viewer

A module that lets you view the status of individual segment requests and, if necessary, cancel them.

required calls

In a routing set, the maximum number of calls that can be offered to a forecast group while still meeting the forecast group's service quality goal.

resolution

The process by which Aspect Workforce determines which state should be reported for a given employee, a given perspective, and a given period of time. This is not always simple, because segments can overlap each other, and more than one segment can occur during the period in question.

roster

A set of perpetual, one-week schedules in which each assigned employee rotates to the next schedule each week. When you use a roster to create trial schedules for a particular week, the specific employee/schedule assignments for that week are determined automatically.

Roster Manager

A module with which you create and maintain rosters and use them to create trial schedules. This module is optional. To enable it, you must set the Roster system parameter to True.

rotation

A Trial Schedule Manager option that lets you select a group of trial schedules and automatically move each assigned employee to the next (or previous) schedule in sequence.

routing set

A table that associates forecast groups with the staff groups to which their contacts are to be routed for handling. Each forecast group’s workload thus serves as the primary basis for calculating required staff for the associated staff group, or groups.

scenario

A set of objectives and assumptions used in forecasting, scheduling, or intra-day performance forecasting.

Scenario modules

The modules you use to create and update scenarios for forecasting and scheduling.

schedule

A list of all the segments for a single employee over a span of days. A schedule has different properties depending on its stage of existence.

Schedule Editor

A tool that you use to add, edit, or delete segments. It is available as two modules: Trial Schedule Editor and Official Schedule Editor.

schedule inflexibility

A phenomenon that tends to create overstaffing in some periods when full coverage is the objective in creating a schedule set. This is caused by the fact that it is impractical to have extremely short schedules for covering momentary peaks in contact volume. To achieve a near-perfect match of staff and workload at all times would require shifts of virtually every length; for example, 2-hour shifts, 45-minute shifts, even 15-minute shifts.

schedule preference

A description of the days and hours that an employee would like to work, used by the automatic assignment process or the preference-based scheduling process to match the employee to a suitable trial schedule.

schedule run

A process that generates schedules; also, the summary information about the run, which is stored for reporting purposes.

schedule sequence

The explicit numbering of the schedules in a schedule set, primarily to provide a default order and a unique means of identification.

schedule set

A group of trial schedules designed to be used together; the “container” that the scheduling process uses to store schedules. In principle, a schedule set represents one approach to satisfying the forecast staffing requirements for a particular range of dates.

Schedule Summary

A module that you use to view summary information about selected employees’ official schedules, especially the times at which each day begins and ends.

schedule tally

A count of the number of staff scheduled to be handling or waiting for work in a given period; that is, the number of staff whose official segments put them in the work state with which that staff group is associated.

schedule template

A set of scheduling parameters that provides the scheduling process with rules and models from which to generate schedules.

schedule template modules

The modules that you use to create and update schedule templates.

schedule test

A process that compares a schedule set to a staffing requirements forecast to calculate an estimate of the service quality that customers will experience; also the summary information about the test, which is stored for reporting purposes.

schedule trade

A transaction in which two employees agree to work each other’s official schedules, or one employee agrees to work the other’s schedule, on a specific date or dates. You can enter or request schedule trades using the Special > Trade option in Segment Worksheet. Employees can request trades and respond to trade requests using WFO.

Schedule Wizard module

A module with which you can generate schedules using a wizard.

Scheduler modules

The scheduling modules you use to generate and test trial schedules.

scheduling

The process of creating a set of work schedules to meet the staffing requirements for a specific range of dates.

scheduling group

An employee group that may be assigned schedule preferences and special scheduling characteristics for use in group scheduling. The schedule preferences are like those used in multiple-schedule assignment.

scheduling scenario

A set of objectives and assumptions used in generating schedules for a particular staff group.

scheduling unit

An Aspect Workforce configuration setting that establishes the smallest block of time that can be used in creating schedules.

scorecard

A table that displays the results of each check (other than personal account checks) that has been performed on a particular segment request. The results are listed vertically by rule and horizontally by date requested. The scorecard is displayed on the Scorecard page of the Request Editor form (Request Manager module).

seasonal factors

A historical pattern consisting of 12 factors, one for each month, that tells Aspect Workforce how much each month can be expected to deviate from the average monthly traffic year after year. For example, a contact center might always have peak months in the spring and early summer and slow months in the late fall and winter. Since the average monthly traffic would be represented by a factor of 1.00, a seasonal factor of .75 means that the month will be 25% slower than average, while a factor of 1.15 means that the month will be 15% busier than average. The sum of the seasonal factors is always 12.

secondary segment

A segment that results in personal accounting transactions for another segment. The personal accounting closeout process debits or credits the primary account for each secondary segment it finds.

security profile

A set of permissions that determines which software features and data a user can access. A given security profile can be assigned to any number of users.

segment

A record of how an employee is spending a specific period of time. Segments are the basic building blocks from which schedules are formed, as well as the essential records that make tracking possible. They can represent either work or non-work (for example, breaks and absences).

segment category

A group of segment codes that have the same basic attributes: for example, work segments or break/lunch segments.

segment entry rule

A rule that you set up to prevent illogical or undesirable relationships among different official segments for the same date—for example, a break that falls outside of the employee's shift.

segment package

A preset combination of segments that you can add for a given employee all at once, as opposed to having to add each segment individually. You create segment packages (which you can use again and again) for typical combinations of segments; for example, overtime shifts with breaks.

segment rank

The relative priority that Aspect Workforce gives to a segment whenever that segment overlaps another (in time) for the same employee. The highest-ranking segment (lowest rank number) is the one that gets counted during the period in question.

segment request

A transaction entered on behalf of (or by) an employee requesting a specific change to the employee's official schedule, such as time off from work, extra work time, schedule adjustments, or trades with other employees. When approved, segment requests become official.

segment window-rule set

A set of rules you create to control the creation and placement of required segments—especially breaks—during schedule generation or break optimization. Window rules are so named because each break has a defined time range, or “window,” in which it can be placed relative to the shift as a whole.

service analysis forecast

A special forecast that calculates the monthly staff required for each point in a range of different service quality values.

service level

The percentage of offered contacts answered within a certain number of hours, minutes, or seconds. This statistic can be used as the basis for calculating required staff.

service quality

A measure of how quickly a forecast group’s contacts are being answered, expressed as either average delay or service level. Service quality is both a statistic and a staffing basis.

shift

On a given day, the period in which an employee is scheduled to be on duty, defined by the earliest and latest segments associated with that date. Also used in reference to the segments themselves, especially container segments.

shift template

A type of schedule template that defines a particular type of shift in terms of its container segment, length (span), time-of-day restrictions, and workday patterns.

shrinkage

The percentage of scheduled work time in which, for various reasons, employees are not actually doing the primary work. In effect, a workforce “shrinks” whenever an employee is not working as scheduled or is working unproductively. Therefore, when calculating staffing requirements it is necessary to “inflate” the ideal number of staff by the same percentage so that you will know how many scheduled staff are needed.

single-schedule assignment

A method of automatic assignment in which each employee is assigned to only one schedule in a given run.

single-skill configuration

An Aspect Workforce configuration that uses the conventional model for routing contacts, in which Aspect Workforce assumes that agents handle contacts from only one source (forecast group) at a time and that each source is routed to only one agent group (staff group). This setting simplifies your use of the software, but it does not allow you to model skill-based routing.

skill

The ability of an employee to handle a particular work state. You define an employee’s skills by entering one or more skill associations.

skill association

An indicator of an employee’s ability to handle a particular work state during a particular calendar period. You enter an employee’s skill association(s) so that Aspect Workforce can match the employee with a suitable schedule.

skill-based routing

An ACD feature that allows a given contact type to be routed to any of several agent groups, and a given agent group to handle a mix of more than one contact type at a time.

span

The total amount of time that elapses from the beginning of a shift to its end, regardless of breaks.

special distribution

An intra-day distribution you create for a day in which contact patterns differ significantly from those normally occurring on that day of the week. A special distribution is a single-day pattern that you can use instead of the normal distribution on any date in a forecast.

special operating hours

The operating hours that are defined for specific calendar periods, during which they override the normal, recurring operating hours.

staff adjustable

In intra-day performance and scheduling calculations, an attribute of a staff group, signifying that the number of required staff can change to accommodate a fixed number of forecast calls. Typically, staff-adjustable staff groups are those whose work is not outsourced. Staff groups may be volume-adjustable or staff-adjustable, but not both.

staff group

The organizational unit for which Aspect Workforce calculates required staff and maintains work schedules, defined as a work state for which agents are needed at a particular location. The number of staff required for a particular group tells you how many agents with that skill association you will need at that location.

Staff perspective

A special perspective that is used to indicate whether or not a particular type of segment entails handling work, and if so, what kind of work.

staffing basis

One of several alternative scenario assumptions you can choose for calculating the number of staff required in each period of the day. For example, you might want to know the number of agents required to achieve a certain service quality or to maximize net revenue.

staffing overlays

A set of staffing requirements, by period-of-day and day-of-week, that are automatically added to the calculated requirements for any dates you specify. You set up overlays by staff group (using Historical Patterns) and apply them as part of a staffing scenario.

staffing requirements forecast

A calculation of the number of agents required in each period of the day to handle the forecast workload for that period.

staffing scenario

A set of objectives and assumptions used in calculating staffing requirements for a particular staff group.

staffing tolerances

A set of upper and lower net staff limits, by period of day and day of week, that you can use for automatically approving and denying segment requests.

Staffing Tolerances module

A module with which you specify the upper and lower net staff limits that certain segment requests are checked against. This module is included with WFO.

start-time interval

A scheduling run-time option that governs the times at which schedules can start; for example, at 15-minute intervals as opposed to 30-minute intervals.

state

The value, or effect, that a segment has with respect to a particular perspective. For example, from the Pay perspective, a Break segment might have a state of either Paid or Unpaid.

stopping rule

A scheduling scenario assumption that defines your overall staffing objective and thus tells the scheduling process when to **stop** generating schedules.

superstate

A user-defined formula that provides Aspect Workforce with a complex method for tallying employee time, based on two or more alternative states from the same perspective or on an intersection of two or more states from different perspectives.

system administrator

An Aspect Workforce user who is authorized to do setup, maintenance, and administrative tasks.

system/customer-initiated

Call work initiated by the ACD or dialer. Can refer to an inbound call routed to an agent by the ACD, or an outbound call initiated by a dialer rather than by an agent. Compare agent-initiated.

talk time

The elapsed time from when an agent answers a contact until the agent disconnects.

Tally Editor

A module that lets you view and edit static staff tallies for selected staff groups and superstates. This module is enabled only when Encompass is enabled.

tally server

An automatic process that monitors official schedules and calculates the number of scheduled staff on duty for each staff group throughout the day.

team schedules

A group of schedules, typically generated from the same master template, that cover the same number of days, have the same days off, and have the same start and stop times. “Subteam” schedules are team schedules that also have identical breaks.

template-based scheduling

An approach to generating trial schedules in which schedules are created based on schedule templates and staffing requirements.

terminated employee

An employee who, as of a certain date, cannot be assigned to a trial schedule, an official segment, or an employee group. The employee is thus effectively deleted, but Aspect Workforce nevertheless preserves all prior records of the employee in order to maintain a valid tracking history.

threshold set

A set of upper and lower percentage limits, by period-of-day and day-of-week, that define acceptable ranges for statistical differences in intra-day performance. Aspect Workforce highlights certain statistics when one statistic exceeds or falls short of another by a percentage outside of the threshold range.

tracking

An Aspect Workforce feature that models actual events and activities in your work center to aid you in short-term planning, daily management, and analysis of staff performance. The tracking functions involve managing employee information, official schedules, and intra-day performance.

tracking date

The date you have chosen as a frame of reference for selecting or viewing tracking information. See also default tracking date.

trade request

A type of schedule request in which one employee asks to exchange (“swap”) official schedules for a given day with another employee. Employees can request trades or respond to trade requests using WFO.

tree

A special list box that displays a set of objects as a multilevel outline based on their hierarchical relationships; also, the group of items that are in such a relationship.

Trial Schedule Editor

A tool for editing the general properties of a trial schedule (including the assigned employee) and for adding, editing, and deleting its segments. Trial Schedule Editor is available from within Trial Schedule Manager.

Trial Schedule Manager

A module that lets you create, edit, and combine schedule sets, assign employees to trial schedules, and create official segments from those schedules.

trial schedule/segments

A schedule that has not been made official and thus may or may not actually be used. It’s called *trial* because it is being *tried*, or tested.

Trial Segment Worksheet

A module that you can use to quickly add, edit, or delete segments in trial schedules and save your changes as one batch.

two-way trade

A schedule trade in which both employees are working each other’s schedules.

understaffing limit

The percentage by which you will allow the scheduling process, when the stopping rule is Full Coverage, to fall short of the required staffing level in any period. This typically provides more economical coverage during the least-busy periods of the day.

unproductive time

The amount of paid time in which employees are scheduled to be handling work with standard efficiency but are actually not. Unproductive time generally involves either working inefficiently or engaging in some non-work activity that doesn’t get tracked (for example, a trip to the water fountain). Aspect Workforce uses a special shrinkage category to model unproductive time.

Updater

An automatic process that validates any pending change, addition, or deletion of official segments and, if the transaction is valid, saves the results to the Aspect Workforce database.

volume adjustable

In intra-day performance and scheduling calculations, an attribute of a staff group, signifying that the number of scheduled staff remains constant while the number of offered calls is adjusted to

accommodate the number of scheduled staff. Typically, volume-adjustable staff groups are those whose work is outsourced. Staff groups may be volume adjustable or staff adjustable, but not both.

volume forecast

A prediction of the offered contact volumes (or other work units) for specific days and/or intra-day periods.

waiting list

A list of segment requests that have not been approved, normally because of insufficient group allowances. Waiting requests can be approved later if more hours become available. Also called *waitlist*.

weighted average

A method of averaging several numbers in which some numbers are increased before averaging because they are more significant.

wizard

A special tool that helps you accomplish a task by asking you questions and guiding you through the necessary decisions. The same task can usually be done using a more direct method, which might provide more options or require more expertise.

work rule

A business rule that employee schedules must not violate under normal circumstances.

work segment

A segment that represents handling work; that is, any segment having a work state.

work state

A state in the Staff perspective that represents handling work; that is, any state with which a staff group has been associated. A segment having such a state indicates that the employee is scheduled to be handling that kind of work.

work type

The basic category into which you group your contacts (or other work), without regard to where the work gets handled.

workload

The total duration of all contacts in a given period (half hour or quarter hour), not counting any time spent in queue. This quantity is equal to the number of contacts times the average handle time per contact.

wrap-up time

The time that an agent spends completing a transaction after the associated contact has already been disconnected.

About Aspect®
Aspect is dedicated to transforming the service economy by humanizing the workforce experience. Their WorkforceOS platform offers a robust workforce management solution that aligns employee preferences with business needs enhancing scheduling, predictive insights, and collaboration tools. Supported by its parent company, Alvaria Inc., which boasts over 50 years of leadership in workforce management technology, Aspect is a trusted partner for large global enterprises across key sectors, including financial services, airlines, automotive, insurance, retail, telecommunications, and utilities. The Aspect WorkforceOS stands out as the only culture-driven WEM software designed to foster work-life balance while maximizing ROI for businesses. For more details, visit www.aspect.com

© Copyright 2025 Alvaria, Inc. All Rights Reserved. 9026US-D 2/23