Oracle Real-Time Scheduler

Configuration Guide

Release 2.2.0.0

E50663-02

July 2014

ORACLE

Oracle Real-Time Scheduler Configuration Guide, Release 2.2.0.0

Contents

Chapter 1

Chapter 2

Chapter 3

Addendum C

Chapter 1

Overview

This guide describes how to configure Oracle Real-Time Scheduler (ORS). The target audience of this configuration guide is implementers and system administrators responsible for configuration and initial setup of the application.

This section provides information about the following:

- Oracle Utilities Application Framework
- Organization of this Guide
- Architecture Overview
- Oracle Utilities Application Framework Configuration Tools
- Entity Naming Conventions
- Custom Algorithms
- Embedded Documentation
- Demonstration Examples
- Recommendations for Creating a Production Environment
- Integrations with Other Oracle Applications

Oracle Utilities Application Framework

Oracle Real-Time Scheduler is based on the Oracle Utilities Application Framework (OUAF). For information about using and configuring basic Framework functions, see the Oracle Utilities Application Framework documentation. This guide only covers configuration of functions specific to Oracle Real-Time Scheduler.

Organization of this Guide

The body of this guide presents conceptual information to help you understand how the system works as well as how the various configuration options affect system functionality. Once you have an understanding of the system's capabilities, you can plan your data setup and design any customizations you want to implement.

When you are ready to implement your design, use the System Setup Quick Reference Table to guide you through the setup process. This reference guide lists each object that can be configured, defines any prerequisites for configuration, and lists objects referenced by or associated to the object being configured.

> **Note:** The sequence in which you configure system objects is very important. The System Setup Quick Reference Table describes data dependencies and

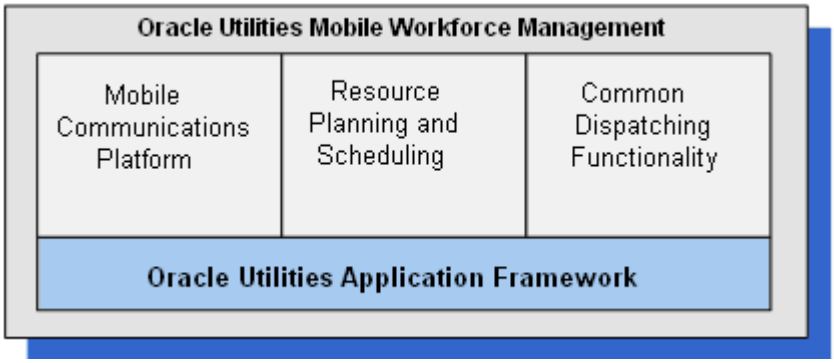
defines the order in which objects should be configured. By following this sequence carefully, you can streamline the configuration process and reduce the amount of time required for setup.

This section includes the following topics:

Architecture Overview

Oracle Real-Time Scheduler simplifies and optimizes the scheduling, dispatching, and tracking of mobile service crews and field activities.

The application is comprised of three functional components which are built on top of the Oracle Utilities Application Framework:



- **Resource Scheduling and Planning:** Supports resource planners and service managers in managing resources, planning shifts, and scheduling work. The system automatically generates shifts and optimizes the schedule based on your business rules. This component manages the following user functions:
  - Resource Management setup and maintenance
  - Service Management setup and maintenance
  - Scheduler setup and maintenance
- **Common Dispatching Functionality:** Supports dispatchers as they handle scheduling exceptions throughout the day, and enables context-based decision making at the dispatcher level. The system can be configured to automatically dispatch all activities or limit auto-dispatching to certain activity types or shifts. The system maintains real-time communication with mobile resources, tracks the location of crews and vehicles, and allows dispatchers to monitor and manage activities, crews, alerts, and key performance indicators. Common dispatching functionality is provided through the **Common Dispatching Interface (CDI)** portal.
- **Mobile Communications Platform:** Provides a user interface on mobile devices to support mobile crews as they perform service. The mobile platform facilitates communication with the dispatcher, provides GPS-based mapping services, and processes activity status updates and work completion details.

Oracle Utilities Application Framework Configuration Tools

The Oracle Utilities Application Framework (OUAF) configuration tools can be used to create and customize system entities, such as business objects, portals, zones and UI maps. Refer to the Oracle Utilities Application Framework configuration tools documentation for instructions on using this tool.

Rather than duplicating concepts and procedures presented in the Oracle Utilities Application Framework configuration tools documentation, this documentation identifies Oracle Real-Time Scheduler-specific objects that can be configured and customized using that tool, as well as application parameters and objects that can be managed within the Oracle Real-Time Scheduler application components themselves.

This guide assumes that all individuals responsible for system configuration and implementation are be familiar with Oracle Utilities Application Framework and will have completed training on this tool.

Entity Naming Conventions

Oracle Real-Time Scheduler uses naming conventions to identify and distinguish entities that belong to different Oracle applications. These conventions can help you locate entities and understand their context.

Each base product prefixes its entities with its 2-character owner code. For example:

- All Oracle Utilities Application Framework entities start with “F1”
- All Oracle Real-Time Scheduler entities start with “M1”
- All Oracle Utilities Mobile Workforce Management entities start with “M2”
- Custom entitles should be prefaced with “CM”

  Oracle recommends that develop your own set of conventions for the entities you create. If you create new entities, DO NOT use these prefixes; use the prefix “CM” to identify entities that have been customized.

Custom Algorithms

Many functions in the system are performed using user-defined algorithms, also referred to as plug-ins. For example, user-defined algorithms can be used to create specific types of alerts and KPIs, generate shifts and POUs, and customize the content that appears in the dispatcher **CDI Scheduling Gantt** zone.

Custom algorithms allow implementers to modify how the system responds to certain system events. The system provides 'plug-in spots' where the custom algorithms can be invoked instead of the base algorithms provided with the system.

Refer to the following for related information:

- For a list of all related system events supported in the base package refer to Appendix B: Algorithm Entities.
- For instructions on creating custom algorithms, refer to the Oracle Utilities Application Framework documentation.
- To view information about specific algorithms provided with the base system, use the Application Viewer (also described in the Framework documentation). The Application Viewer provides information about the base logic, inputs, and outputs of each algorithm entity or plug-in spot.

Embedded Documentation

As with all Oracle Utilities Application Framework applications, extensive internal documentation is available:

- **System Objects** - Detailed descriptions of system objects are included in the maintenance portals for those objects.
- **Lifecycles** - The lifecycle of each business object is described on the **Lifecycle** tab and depicted in flow diagrams on the **Summary** tab. This information is extremely useful for implementers and system administrators.
- **Functional Help** - Embedded help is provided for all non-obvious fields at the UI map section level. The help content can be customized during implementation by adding custom help to the **Override Help Text** attribute. If a field on a UI map has associated help text, a **?** icon appears next to the field when the UI map is displayed. When the user clicks this icon, the contents of the **Help Text** attribute (or the **Override Help Text** attribute, if populated) is displayed in a pop-up window.

Demonstration Examples

The demonstration environment shipped with the base product provides setup examples that may be helpful as you implement your system.

Recommendations for Creating a Production Environment

Oracle recommends that you do not clone the demonstration environment as a basis for a new production environment. The demonstration environment typically includes transactional data that will be irrelevant to your production environment and can cause unexpected issues if it is not purged correctly. The recommended process is to start a new production environment from a new installation and migrate "clean" system data (such as business objects and algorithms) and administrative data (such as sample activity types or other administrative entities) from the demonstration and/or test or development environments as applicable.

Your implementation can use bundling and/or the Configuration Migration Assistant to move system and administrative data. Instructions for using these tools are provided in detail in the chapters titled "Bundling" and "Configuration Migration Assistant" in the *Oracle Utilities Application Framework Administration Guide*. For more information about base package Oracle Real-Time Scheduler migration requests that can be used when migrating administrative data, refer to Appendix D: Configuration Migration Assistant (CMA).

Please contact Oracle customer support for assistance.

Integrations with Other Oracle Applications

Refer to the appropriate Oracle Utilities Integration documentation for information about specific integrations.

Chapter 2

General Component Configuration

This section describes configuration of general components, including the following:

- Installation Options
- Feature Configuration
- Master Configuration

Installation Options

Installation options define the individual applications installed on your system and identify algorithms used to implement core system functions. These options also define global parameters such as the administrative menu style (alphabetical or functional), the country, language, currency code and the base time zone to use for this implementation. Take note of the following details regarding installation options:

- Installation options are stored in the installation record for your system.
- Navigate to **Admin > Installation Options > Framework** to configure these options. This portal is part of Oracle Utilities Application Framework and is described in detail in the Framework documentation.

Time Zones

This section describes how time zones are stored, displayed and entered throughout the system.

Base Time Zone

Server time zone, database time zone and the time zone specified on **Admin** -> **Installation Options** must match. This establishes the **Base Time Zone** which represents the system clock. If these are not in sync, the system does not function properly.

The date/time attributes for all time-sensitive application entities, including tasks and shifts, are stored in the server application time zone in standard time. This means they are independent of any seasonal time shifting adjustments, if applicable, in that time zone.

The system also allows data to be entered and displayed in a different time zone in legal time. This means that it is adjusted for seasonal time shifting, managing the conversion back and forth between the data entry and the storage time zones.

Entity and User Time Zone

If it is necessary to show a different time zone on some system entities, then the time zone can be set on those entities or in user preferences. For example, entities associated with a geographic location such as activities, crew shifts and meetings, are entered and displayed in their

corresponding time zone. The rest of the application shows the user's preferred time zone to display date and time information.

This Entity Time Zone is used for data entry and display purposes. When entity time zones are entered, the system converts it to the base time zone before storing it in the database. Likewise, the system converts data from base time zone to the entity time zone before the time is displayed on a record.

If an entity is not associated with a specific time zone data entry is made using the time zone specified on the user record. Again, this is just for data entry and display purposes.

The **CDI** portal displays data according to the user time zone, however, users have the ability to toggle time zones as needed.

Daylight Saving Time

The term "Standard time" refers to the time as if no daylight saving time (DST) is applied. This virtual time line has no gap hour or duplicate hour, it is continuous as if DST shifting was never used. The term "Legal time" refers to actual clock time as impacted by DST. This time line includes a gap hour on entry to the DST period and a duplicate hour on DST exit. All date/time information is *stored* throughout the system in base time zone standard time, but all date/time information is *entered and displayed* in the UI in entity/user time zone legal time. The same display vs. storage rules apply on mobile devices. Crews using the mobile application (MCP) work their schedule using their local device time zone and the system handles the conversion of the data to and from the server time zone.

Time zone and DST conversions are online user interface features. To the user, data is entered and displayed in entity/user time zone legal time as it would be on any local clock they are using. The user should not be aware that it is actually stored in the database in base time zone standard.

Time Zones and the Activity Template Generator Tool

Data in the .csv file is assumed to be in base time zone standard, as if stored in the database. Since the user interface for the tool does not complete and shifting, time is added as is, in base standard time. When an activity is displayed on portals it is displayed the entity time zone legal as described above.

Installation Algorithms

Installation algorithms implement global system functions and can be customized for each implementation.

The base package supports the following installation options related to system events:

- **Address Information**: Responsible for formatting address information for display throughout the system on the server.
- **Address Geocoding**: Responsible for geocoding an address (converting an address to a geocode longitude/latitude pair).
- **SMS Send**: Responsible for processing outgoing SMS messages.
- **SMS Receive**: Responsible for processing incoming SMS messages sent from a crew.
- **Remote Script Invocation**: Responsible for processing remote messages sent from the mobile device.
- **To Do Pre-Creation:** Responsible for making To Do Entries searchable by linking them to their related entity via a characteristic.
- **Determine Connected Device:** Responsible for assigning a virtual MDT when using the mobile application in connected mode.

Refer to Appendix B: Algorithm Entities for a complete list of system events supported by the base package.

Feature Configuration

Oracle Real-Time Scheduler includes 'specialty' zones that invoke an external URL to render their contents. The **Gantt** and **Calendar** zones are both specialty zones. The functionality provided by these zones requires that you set up a feature configuration of type "Application Context Root," which defines the root location of the main application and other related applications.

Refer to Appendix A: System Setup Quick Reference Table to determine when in the configuration process you should perform feature configuration.

Follow the embedded help instructions associated with each feature configuration type to fully set up the associated option.

Map Server Configuration

The **Common Dispatcher Interface** (CDI) supports a map view of the dispatcher's monitored area. This CDI Map functionality requires that you set up a feature configuration of type "Map Server Configuration."

Master Configuration

Master Configuration is another source of global parameter records used by a system implementation.

Oracle Real-Time Scheduler uses a Global Configuration record that controls core system functions. This record must be set up for the system to properly operate. Refer to Appendix A: System Setup Quick Reference Table for more information on when to set up this record.

Refer to the embedded help for descriptions of the settings on the Master Configuration page.

Planning Horizon

The planning horizon, defined in the Global Configuration, is the number of days into the future to plan for crew shifts and periods of unavailability. This setting controls the number of day's worth of shifts to generate for perpetual crew shift templates. For example, if the planning horizon is set to 60, the system generates crew shifts and periods of unavailability (POU) sixty days into the future. Generation occurs in a sliding window; each day generates data for the last day incrementally. At any given time, the number of days of planned shifts and periods of unavailability is equal to the planning horizon value.

The planning horizon should be equal to or greater than the scheduling horizon, which is defined for a particular scheduler configuration. The scheduling horizon is the number of days into the future the scheduler will consider when scheduling crew shifts and activities. The scheduling horizon designed to limit the scope of data actually being scheduled by a scheduler. Oracle recommends that the scheduling horizon be kept as short as is practical while still meeting customer business requirements. The optimum value for the scheduling horizon strikes a balance between long term planning and short term flexibility.

> Refer to the Scheduler Configuration chapter for more information about configuring schedulers.

Start Weekday Option

The Start Weekday Option in the Global Configuration allows you to specify which weekday marks the start of a week for your organization. The system uses this to properly display weekly-based information, such as shift weekly templates and calendars.

Chapter 3

Resource Management Configuration

This section provides configuration information for entities related to resource management including:

Locations

You can define a set of common locations, such as service centers or business offices, that can be selected and referenced later.

Predefined locations can be used as follows

- Logon and logoff service center locations for crew shifts
- Meeting locations for crew, also known as a crew periods of unavailability
- Locations where crews restock their vehicles
- Depot locations

A location record defines the name and description for the location, the physical address, and the geocoordinates. The location address must be geocoded so the scheduler can direct crews to and from a common location.

The location business object provided with the base package is associated with an algorithm that calls the address geocoding service (defined on the installation options record) to geocode the location address (convert the address into a longitude/latitude pair), requesting the highest geocoding quality level. Refer to Appendix B: Algorithm Entities for more information about the address geocoding algorithm service on installation options.

Skills

A skill indicates the ability of a mobile worker to perform certain types of work. Mobile workers can have skills; mobile worker types can have assumed skills; and activity types can require skills.

A skill record defines the skill type and description, and indicates whether or not the skill has an effective period, such as a licensed-based skill. If a skill has an effective period, you will be required to enter the effective dates when you associate the skill with a mobile worker.

A skill record may represent a certain level of expertise within a skill set. Skill levels are used to make sure that skilled resources are being used as efficiently as possible. Schedules can assign crew members to an activity based on their level of skill so as not to waste more experienced resources on simple task, or to be sure to have a worker with a high level of skill where needed.

Review any skills and skill levels needed to support your business requirements, as needed.

Equipment

Equipment is any device or apparatus used to perform certain types of work. Vehicles can carry pieces of equipment needed to perform certain types of work; vehicle types can have assumed equipment; activity types can require equipment.

An equipment record defines the equipment type and description. If equipment has an effective period, you can enter the effective dates when you associate the equipment with the vehicle.

Mobile Worker Types

Mobile worker type records are like job descriptions. They define the skills that mobile workers of a particular type are assumed to have. Assumed skills can be overridden for an individual mobile worker.

The mobile worker type definition also controls what business object is used when creating a new mobile worker of that type. Thus, it dictates the business rules governing content and behavior common to all mobile workers of this type.

Mobile Workers

A mobile worker is an individual workforce resource that performs work as part of a crew. Mobile workers can have skills indicating their ability to perform certain types of work.

A mobile worker record defines the mobile worker type, the skills possessed by the worker (which default to the assumed skills of the mobile worker type), the service areas in which the mobile worker typically works, and other attributes. You can also specify the Relative Efficiency in the scheduler. This specifies the mobile worker's efficiency relative to expected efficiency for mobile workers of this type.

> Refer to Entity Parameters in Addendum C: Scheduler Parameters for more information about scheduling factors.

A mobile worker's home address can be used as a shift logon or logoff location, if the mobile worker is the only person on that crew. In this situation, the home address must be geocoded. For

this reason, the mobile worker business object provided with the base package is configured with an algorithm to geocode the mobile worker's address and request an exact match on the address.

> Refer to Address Geocoding in Appendix B: Algorithm Entities for more information about this algorithm.

If your company uses contractor mobile workers, their user record and their mobile worker record needs to reference the contracting company they work for. Refer to Contractor Management Basics for more information.

As a best practice you may want to set the home page for mobile workers to the My Calendar portal. This allows them to view their shift schedule and any planned leaves on a calendar view.

Vehicle Types

Vehicle type records define the equipment that vehicles of a particular type are assumed to have. Assumed equipment can be overridden for an individual vehicle.

The vehicle type definition also controls what business object is used when creating a new vehicle of that type. Thus, it dictates the business rules governing content and behavior common to all vehicles of this type.

If your operations involve transfer of goods such that the load on board a vehicle has to consider vehicle capacity limits, use the depot related vehicle type business object.

Vehicles

A vehicle is a workforce resource used by a crew to perform work. Vehicles can carry pieces of equipment needed to perform certain types of work.

A vehicle record defines the vehicle type and the equipment possessed by the vehicle (which defaults to the assumed equipment of the vehicle type). You can also specify the Relative Speed, in the scheduler. This defines the vehicle's speed relative to the expected speed for vehicles of this type.

> Refer to Entity Parameters in Addendum C: Scheduler Parameters for more information about scheduling factors.

If your company uses contractor owned vehicles, their record needs to reference the owning contracting company. Refer to Contractor Management Basics for more information.

Crew Types

A crew type record defines a particular type of crew, such as a one-person crew or a multi-person crew. It also controls what business object is used when creating a new crew of that type, thus dictating the business rules that govern content and behavior common to all crews of this type.

Crews

A crew is a uniquely named group of mobile workers and vehicles that perform work in shifts. A crew shift is a planned period of time in which a crew (one or more mobile workers and vehicles) is scheduled to perform work. The specific mobile workers and vehicles allocated to a crew may vary from one shift to another, and a crew may perform different types of work and cover different service areas on different shifts. Therefore, crew allocation is defined for crew shifts, not for crews. Likewise, activities are scheduled to crew shifts, not to crews.

If the resources allocated to a crew are typically fixed to a single mobile worker and their specific vehicle you may want to use the single person crew business object that allows you to define these resources at the crew level. This information is used as a default resource allocation when a shift is created either adhoc online or by the shift generation process.

If your single person crews are associated with specific depots to perform work use should use the depot related single person crew business object.

> **Note**: Logic to default resource allocation and any other shift related information from the crew onto a new shift when it is created resides in a plug-in. Refer to Prepare Shift Data in Appendix B: Algorithm Entities for more information about this algorithm.

If your company uses contractor crews, their record needs to reference the contracting company they are associated with. Refer to Contractor Management Basics for more information.

Your implementation might use crew hierarchy to associate crews with a reporting hierarchy. Refer to Hierarchies Basics for more information.

Dispatcher Types

A dispatcher type record defines a particular type of dispatcher. It mainly controls what business object is used when creating a new dispatcher of that type, thus dictating the business rules that govern content and behavior common to all dispatchers of this type.

Depending on your organization needs, you may need only one or very few dispatcher type definitions.

Dispatchers

A dispatcher is a user who manages day-to-day field operations.

A dispatcher record defines the dispatcher type, default date range for monitoring, dispatcher areas allowed to be monitored, and KPIs to be monitored. It also specifies whether or not the dispatcher can change certain settings when logging on to a shift.

Crew Shifts

This section describes key concepts and configuration information related to crew shifts. Refer to online help and the user guide for more information on how to set up shifts and templates.

Crew Shift vs. Crew Shift Type

A crew shift, sometimes referred to simply as a shift, is a planned period of time in which a crew (one or more mobile workers) is available to work. This should not to be confused with the common meaning of 'shift,' which typically refers to the time of day during which a crew performs work, such as first shift or second shift. A crew shift refers to a specific date and time planned for a specific crew to perform work.

Every crew shift is associated with a crew shift type (such as first shift, second shift, etc.) that controls what business object is used when creating a new crew shift of that type. Thus, it dictates the business rules governing content and behavior common to all shifts of this type.

Scheduling Details are Defined on the Shift

The specific mobile workers and vehicles allocated to a crew may vary from one shift to another. In the same way, a crew may perform different types of work and cover different service areas on different shifts. A crew may also start and end their route in different locations on different shifts. Therefore, crew resource allocation and all other scheduling information are defined for crew shifts, not for crews. Likewise, activities are scheduled to crew shifts, not to crews.

Crew Capabilities

The scheduler is encouraged to assign work to crew shifts where the crew possesses the capabilities (skills and equipment) necessary to perform the work. The overall skills and equipment associated with a crew on a specific shift are derived from the resources comprising that crew on that specific shift.

When a crew starts a shift, the shift's resource allocation may differ from its planned allocation. Resource allocation may even change throughout the shift. These changes are communicated to the scheduler, which may reconsider the planned work for the shift and rebuild the shift schedule.

Once the shift has started, any change made to its crew's resource allocation is recorded for reporting and other business needs.

> **Note**: A crew and any of its allocated resources can only be part of one shift at any given point in time. This means you cannot plan shifts for the same crew or mobile worker or vehicle to overlap in time.

Controlling Work Assignment

While a crew must be appropriately skilled and equipped to be assigned certain work, you can further plan and affect the work scheduled to a crew on a specific shift to meet your business needs using service classes.

Consider the following options when designing service classes:

Primary Function

Types of work may be classified by service class. You can designate one of these service classes as the primary function of the crew on a specific shift. This encourages the scheduler to prefer work of that class, if any, over other classes of work. You can also list specific service classes of work that the crew should not be assigned on that shift, regardless of whether they are capable to perform that work.

Reserve Capacity

You can also reserve a certain percentage of the shift time for a specific service class of work, up until a specified lead time before the shift starts. Once started, the reserved time is released and made available to all other classes or work. For example, consider that your business is entering a seasonal busy cycle, and you want to reserve more time for emergency work during this period.

Not Allowed Work

You can also list service classes to instruct the scheduler to not assign this type of work to the crew on a specific shift.

Crew Shift Logon / Logoff

A crew shift must either reference a service center location record for their logon / logoff location or indicate it is the mobile worker's home. The latter is only applicable when the shift is for a single worker. Other than the home option the shift cannot reference an address and has to reference a common location record for the address.

This information is used by the scheduler to properly plan the schedule or tasks assigned to the crew on that specific shift.

Shift Cost Profiles

Shift cost profiles are predefined shift-based cost factors that are used as multipliers against global costs. You can assign a shift cost profile to one or more shift templates or to individual shifts. Each shift is associated with a single shift cost profile.

The following scheduling cost factors are defined in the shift cost profile and apply to all shifts associated with that profile:

- Shift Cost
- Relative Travel Time Cost (Time Cost)
- Overtime Cost
- Entity Parameters: Shift Distance Cost
- Entity Parameters: Cost Idle

> Refer to Entity Parameters in Addendum C: Scheduler Parameters for more information about scheduling factors.

The shift cost profile record defines the cost factor values and may include a list of associated service areas and/or service classes. These associations do not restrict the use of the shift cost profile in any way; you can associate any shift cost profile to any shift. Rather, the system uses these associations to recommend appropriate shift cost profiles when you are defining crew shifts.

Before you define shift cost profiles, study your organization's cost requirements and how they relate to specific service areas and/or service classes.

Drip Horizon

Once a crew starts their shift, you can configure the scheduler to automatically dispatch to the crew all or a subset of their work without requiring dispatcher intervention. The number of tasks to dispatch at any point in time, referred to as the drip horizon, may be controlled on the crew shift itself. Refer to Understanding Auto Dispatch for more information.

Generate Shifts Using Templates

While crew shifts can always be manually created to address a specific situation, they are typically planned and generated ahead of time based on recurring templates. Refer to the Shift Weekly Templates section for more information.

Shift Weekly Templates

This section describes key concepts and configuration information related to shift weekly templates.

What is Shift Weekly Template?

Typically, crew shifts are generated ahead of time from templates that define a cyclical weekly pattern. Resource planners can create shift weekly templates for specific crews, or they can create common templates and subscribe multiple crews to one template. Other attributes defined on both types of templates include effective period, and number of weeks in rotation.

The details for each shift within a cycle are captured on a shift template record. In other words, a shift weekly template is made of one or more shift templates that describe the recurring shift patterns. It is important to note that any detail you can define for an actual shift, you can also predefine on a shift template. Refer to Actual vs. Template Shifts for more information.

Once you have defined your templates, you must apply them so that actual shifts will be generated from them, based on the template's effective dates and your system's Planning Horizon. For example, you might define a weekly template for Crew A in which the crew works its regular 9-5 shift on M-F every week and also works an on-call shift on Saturday and Sunday every third week. Thus, the shift weekly template would rotate every 3 weeks.

Once applied, the system continues generating shifts to meet the planning horizon. Refer to the Shift Weekly Template Monitor section for more information on this background process.

Actual vs. Template Shifts

Regardless of whether it is general or common, a shift template defines the details needed to generate actual shifts based on that template. This means the shift template's business objects has to share a common data structure with the corresponding business object used to create actual crew shifts based on that template.

A shift template is implemented as a crew shift entity that is simply marked as a template rather than an actual shift. Sharing the same maintenance object makes it easier for template and actual shifts to capture same information.

As mentioned earlier, Crew Shift Type defines the business object used to create actual shifts of this type. It also specifies the corresponding business object used to create shift templates of that type. This means a shift template can only create shifts of its same type.

Planning Breaks

When you define crew shift templates, you can plan one or more break types for the shifts. When the shifts are generated, the system automatically creates breaks and incorporates them into the crew shift schedule.

Refer to Breaks for more information.

Planning Non-Productive Tasks

When you define crew shift templates, you can plan one or more non-productive task types for the shifts. When the shifts are generated, the system automatically creates non-productive tasks and incorporates them into the crew shift schedule. The location of the non-productive task is taken from the crew record at shift generation time.

Refer to Non-Productive Tasks for more information.

Shift Weekly Template Monitor

The Shift Weekly Template Monitor batch process controls how often shifts are generated going forward to meet the planning horizon, and thus how many days' worth of future shifts will exist at any time. For example, you can choose to generate shifts once a week, once a day, or multiple times in a day. If you choose to generate shifts once a week, then the system will generate seven more days' worth of shifts each week to meet the planning horizon; if you choose to generate shifts once a day, the system will generate one more day's worth of shifts each day.

Review the default settings for the monitor process, and schedule shift generation as often as appropriate to meet your organization's needs.

Shift Generation Algorithm

The logic used to generate shifts from templates resides in an algorithm associated with the shift template business object. This algorithm is invoked when the shift weekly template to which the shift template belongs is initially applied and also by the shift weekly template monitor process on an ongoing basis. The algorithm is also called if you make changes to a shift template, for which the associated shift weekly template has already been applied, in order to adjust future generated shifts to match recent changes.

Review the logic defined in the base algorithm to determine if it is adequate for your needs.

Refer to Generate Crew Shifts in Appendix B: Algorithm Entities for more information.

Periods of Unavailability

This section describes key concepts and configuration information related to periods of unavailability (POUs).

What is a POU?

A POU is a period of time during which one or more resources are not available to perform work.

POUs might be configured to handle the following types of non-work related situations:

- meetings
- training
- jury duty
- vacation time

- other leave

A POU is of a specific type. All resources associated with a specific POU must be of the same resource class as defined on the POU type.

POU type also controls what business object is used when creating a new POU of that type. Thus, it dictates the business rules governing content and behavior common to all POUs of this type.

The base product supports a crew POU to which multiple crews can be associated as well as an individual resource's POU such as mobile worker, dispatcher and vehicle.

What is a POU Task?

A POU task is used to track a crew's attendance of a POU event. It is only applicable to crew POUs.

Refer to POU Task Types for information about configuring task types.

Recurring POUs

If the type of event reoccurs (such as daily, weekly, monthly or yearly meetings), POU details are captured on a POU template record, which defines how often the event occurs and other attributes.

When a POU template is initially saved, the system generates POUs from the template automatically. The system only generates POUs to meet the Planning Horizon. The Period of Unavailability Monitor is responsible for generating more POUs going forward.

You can only create recurring POUs if the POU Type is defined to support POU templates. Refer to the next section for more information about POU templates.

Actual vs. Template POUs

A POU template defines the details needed to generate actual POUs based on that template. This means the POU template's business objects has to share a common data structure with the corresponding business object used to create actual POUs based on that template. The **POU Template** business object includes elements to support various reoccurrence patterns.

A POU template is implemented as a POU entity that is simply marked as a template rather than an actual POU. Sharing the same maintenance object makes it easier for template and actual POUs to capture same information.

As mentioned earlier, POU Type defines the business object used to create actual POUs of this type. It also specifies the corresponding business object used to create POU templates of that type. This means a POU template can only create POUs of its same type.

Period of Unavailability Monitor

The Period of Unavailability Monitor batch process controls how often POUs are generated going forward to meet the planning horizon, and thus how many days' worth of future POUs will exist at any time. For example, you can choose to generate POUs once a week or once a day. If you choose to generate POUs once a week, then the system will generate seven more days' worth of POUs each week to meet the planning horizon; if you choose to generate POUs once a day, the system will generate one more day's worth of POUs each day.

Review the default settings for the monitor process, and schedule POU generation as often as appropriate to meet your organization's needs.

POU Generation Algorithm

The logic used to generate POUs from templates resides in an algorithm associated with the POU template business object. This algorithm is invoked when the POU template is initially saved and also by the POU template monitor process on an ongoing basis. The algorithm is also called if you make changes to a POU, so the system can adjust future generated POUs to match recent changes.

Review the logic defined in the base algorithm to determine if it is adequate for your needs.

> Refer to Generate POUs in Appendix B: Algorithm Entities for more information.

Breaks

A break is a planned time period within a crew shift during which the crew is eligible for taking a break. During a break, the crew may not carry out work or travel. Unlike a crew-related period of unavailability that takes place in a specific location, a break does not have a defined location.

A break type defines the duration of a break and the duration of the break window, which is the period of time during which the break can be taken. For example, a 15-minute morning break might have a 60-minute window during which the break can be taken.

While you can manually add a break to an existing crew shift, breaks are typically incorporated into planned shifts at shift template design time.

> Refer to Shift Weekly Templates for more information.

Non-Productive Tasks

A non-productive task is a planned time period within a crew shift where a crew can take time to perform a non work related task such as restocking the truck. Non-productive tasks are always associated to a location, and the crew cannot carry out work or travel during this allocated time period.

A non-productive task type defines the duration of a non-productive task and the duration of the non-productive task window, which is the period of time during which the non-productive task can be taken. It also defines an offset which determines how much time into the shift that the non-productive task can take place. For example, a 15-minute morning non-productive task might have a 60-minute window during which the non-productive task can be taken and it can only occur after the crew has been working for at least 2 hours.

While you can manually add a non-productive task to an existing crew shift, non-productive tasks are typically incorporated into planned shifts at shift template design time.

On the template, the user only defines the task type and the earliest time the NPT should be scheduled after shift start. The NPT location is defaulted from the crew record.

Chapter 4

Service Management Configuration

This section provides configuration information for entities related to service management, including information about:

- Remark Types
- Work Profile
- Appointment Booking Groups
- Activity Priority
- Service Areas
- Activity Types
- POU Task Types
- Service Classes
- Dispatch Areas
- Alerts
- KPIs

Remark Types

Standard remarks define different types of comments that can be entered when an activity is completed. Remark types are associated to activity types, indicating the types of remarks that can be entered when completing this type of activity.

Remark types can define custom common indications used by the system itself and/or common remark types known to the host system and sent as part of completion details.

Work Profile

Some activities include a requirement to be completed during certain days or times. For example, cutting a meter for non-payment is not allowed on weekends or hydrant flushes can only be performed during the graveyard shift.

You should configure work profiles as appropriate to typical work patters and to restrict activity types according to regulatory requirements or other requirements.

The system allows you to establish a work profile for activities, so that regulatory restrictions or other limitations or conditions that apply for the activity can be scheduled and managed accordingly (e.g. business hours, off hours, evening, early morning, not weekends, etc.).

With a work profile properly defined and referenced on an activity type, the system ensures that activities of this type are not scheduled outside of the established allowed time windows.

This information is used by the **Update Time Windows** algorithm entity associated with the activity business object. Refer to Appendix B: Algorithm Entities for more information.

To configure work profiles:

- Create work profiles according to your business requirements.

Review activity types to see where work profiles are needed and associate a work profile as necessary.

Appointment Booking Groups

Appointment booking refers to how the system handles requests from the host system for appointments and how it responds with available appointment windows. Appointment booking groups (ABGs) define one or more valid time windows for appointment booking. ABGs are associated with activity types, indicating the ABGs to use when processing appointment requests for activities of that type. Refer to the Appointment Booking section in Chapter 19: Web Servicesfor more information.

Activity Priority

This section describes key concepts and configuration information related to activity prioritization.

Scheduling Priority

The scheduler prioritizes activities based on their scheduling priority, which is a numerical value from 0.01 to 10 that establishes the importance of the activity relative to other activities. 10 is the highest priority.

Activities with a higher priority are scheduled vs. not scheduled, scheduled to an earlier shift, or scheduled earlier in the shift.

If Scheduling Priority is not specified for an individual activity, it is defaulted from the activity type.

An activity may be associated with a **Business Priority Number** which captures the business value of the activity such as cost or other currency amount, number of customers, relative customer importance etc., and is used to derive the scheduling priority number. This is a way to derive the scheduling priority number based on real business numbers. The conversion from a business priority number to a scheduling priority is configured in the **Priority** section on the activity type.

For example, collection activities can be prioritized highest overdue amount first. The host system would include a business priority number equal to the amount overdue. Given this, the activity type logic would derive the scheduling priority based on ranges of amounts overdue:

- Up to $100 converts to a scheduling Priority of 1
- Up to $1000 converts to a scheduling priority of 5
- Up to $999,999,999 might convert to a scheduling priority of 10
- (with other ranges in between)

Another example is for the host to include a business priority number equal to the number of customers impacted so that the activity that impacts the most customers is prioritized higher.

If your organization needs to maintain the concept of priority based on activity type but allow activities of same type to prioritize differently, then it is important to keep scheduling priority ranges allocated to specific types of work and let the stepped list be a further list of ranges within the range designated for the activity type. This way the scheduling priority will represent the preference between one type of work to other types as well as the preference within activities of one type of work.

Note that on the activity type, a scheduling priority can be defined and ranges can be defined. When determining priority, the system attempts to use the ranges first, based on information from the activity, however if the priority cannot be determined from the range, the value in the **Scheduling Priority** field is used.

When requests to add or update activities are sent from the host system, the business priority and scheduling priority can be included as part of the request. This also applies for appointment booking requests. If both values are included, the scheduling priority overrides the business priority number.

Priority Profile

The scheduler applies a factor onto the activity's scheduling priority to calculate its priority over time. This factor is defined by a Priority Profile as a function over the activity's broad time window.

The priority profile defines the following attributes:

- The sequence in which activities will be displayed in application screens.
- Whether or not work is mandatory or optional. If mandatory the activity will be scheduled by its due date regardless of cost. If optional, scheduling the activity is considered optional and is driven by costs.
  Note that emergencies are always considered mandatory. You can override this on the activity.
- The function that the scheduler uses to calculate a factor applied to the activity's scheduling priority over time to make it stronger, weaker or unchanged as needed. For example, short-cycle, SLA-oriented work is critical at the beginning of the window and diminishes after that. Long-cycle work, such as meter inspections, is low priority in the beginning of the window and becomes critical toward the end.

  Note: The meaning of the priority function defined by a priority profile has changed: Prior to release 2.2.0 it was a factor over the allocation priority cost only which was applicable to optional work only. As of release 2.2.0, it is a factor that applies over an activity's scheduling priority over time whether it is mandatory or not. Also, mandatory work for which the time-windows span more than one day, is considered optional until the activity's end time window is within the horizon.

Service Areas

A service area defines the logical boundary of an organization's territory. Typically, not all crews are allocated to do work in the whole territory and not all dispatchers monitor the whole territory. Service areas are used to divide a territory into broad areas of service.

Service areas can be based on geographic regions, services provided, business units, or any other breakdown that is meaningful for your organization.

Mobile workers may typically work in specific service areas, which comprise their base service areas. Crew shifts may be limited to working in specific service areas, and dispatchers may be responsible for monitoring specific service areas. Each activity is associated with a single service area. If it is not explicitly provided by the host system, the rules to determine an activity's service

area reside in a plug-in on the activity business object so they can be customized as needed. Refer to the Determine Service Area algorithm for more information.

Service areas can also be used to restrict the crews and activities monitored by a dispatcher. Refer to Dispatch Areas for more information.

If your company uses external contractors your company may offer them a certain capacity of work in a specific service area and of a specific class of work. Refer to Contractor Management Basics for more information.

Your implementation might use hierarchies to group service areas by territories, regions, offices, etc. Refer to Hierarchies Basics for more information.

Geographic Boundaries

You can capture the actual geographic boundaries of a service area as part of the service area record. If a service area is defined by postal codes and if you can obtain the postal code boundary data (set of geocodes), then these can be loaded to the service area record.

To capture service area boundaries by postal code:

- Indicate in the map viewer feature configuration that you have licensed Oracle Spatial
- Upload your postal code boundary data to the **M1_POSTAL_CODE_BOUNDARY** table
- Create your service area records
- For each service area, navigate to the **Map** tab and click **Edit Text**. List the postal codes within the area comma delimited and save.

  Note: If you are upgrading from a previous release and have already set up the "Postal Code to Service Area Mapping" extendable lookup you may use the "Convert Postal Codes to Service Area Boundary (M1-CnvPosCd)" BPA script to convert your existing mapping data into service area boundaries. You no longer need to maintain the extendable lookup mapping.

If the above is not an option for your organization, geographic boundaries for a service area may be drawn in an external Geographic Information System (GIS) tool and the resulting ordinates can then be copied and uploaded to the system by using the **Map** tab, clicking **Edit Text** button and pasting the coordinates as the **Polygon Definition**. Ordinates must be entered as comma delimited in longitude, latitude order.

If your company provides multiple types of services and divides the service territory differently for each type of service, you should indicate the **Service Type** associated with each service area. The system validates that service areas for the same service type do not overlap geographically.

If you have uploaded the geographic boundaries of your service areas

- The base "Determine Service Area" algorithm uses this information to determine the service area covering the activity's location geocode.
- The **CDI** map can show the dispatcher's service areas
- The **Scheduler Area** portal shows the covered service areas on the **Map** tab.

Postal Code to Service Area Mappings

If your service areas are grouped by postal codes, but you cannot capture the geographic boundaries of your service areas you can use the base "Determine Service Area" algorithm "Postal Code to Service Area Mapping" extendable lookup to map a postal code to a service area.

For an easier setup you can use the "Upload Postal Codes for Service Territory (M1-UpPostCd)" BPA script to upload these mapping records from an excel spreadsheet.

As mentioned earlier, if your company provides multiple types of services and divides the service territory differently for each type of service, you should indicate the **Service Type** associated with each service area. In addition you should use a separate extendable lookup for each service type as follows

- Subclass the base extendable lookup for each service type supported
- Specify the service type as a BO option
- Use a separate excel spreadsheet for each service type
- When using the upload BPA script, select the appropriate extendable lookup for the service type mapping you currently upload

Activity Types

This section describes key concepts and configuration information related to activity types.

Basic Concepts

This section describes concepts that are important to understand before you configure activity types.

What is an Activity?

An activity is a task, typically requested by a host system, to be performed at a particular geographic location. An activity may have a site address and a service address. Only the site address is used by the scheduler to direct the crew to the activity location.

An **activity type** defines skills and equipment required to work activities of this type as well as details about how the activities should be dispatched, prioritized, and scheduled.

What is an Assignment?

An **assignment** is a copy of an activity that is created when an activity is dispatched and, for MDT crews, is also sent to the assigned crew for recording status updates and completion details.

If changes are made on the server to an activity, the system determines which changes are relevant to the crew and sends updated information to the crew accordingly. Thus, the assignment is kept in synch with its original activity. Likewise, when changes are made on the mobile, they are sent to the server immediately (or when a connection is available).

If your company uses contractor management functionality and assigns work to external contractors towards agreed upon capacity, an assignment is also created to represent the contractor's commitment to perform an activity. Refer to Contractor Management Basics for more information.

Activity and Assignment Lifecycles

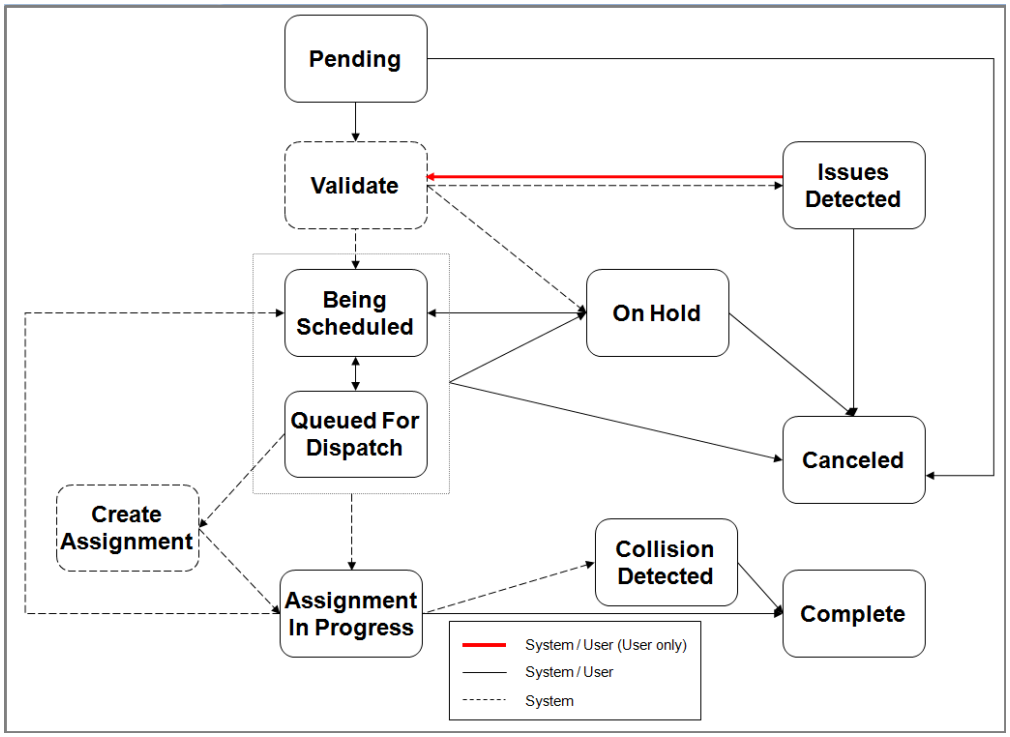
An assignment is created when the activity lifecycle reaches a point in which work should be dispatched to the assigned crew. The lifecycle of an assignment affects the lifecycle of its activity. For example:

- When an assignment is completed, the original activity is automatically completed, provided that one and only one finalized assignment is received for the activity.
- If an activity's current assignment is returned and no other assignment is complete, the activity is rescheduled.
- If all assignments are final and only one is complete, then the activity is completed.
- If all assignments are final and more than one is complete, the activity's status is set to collision detected.

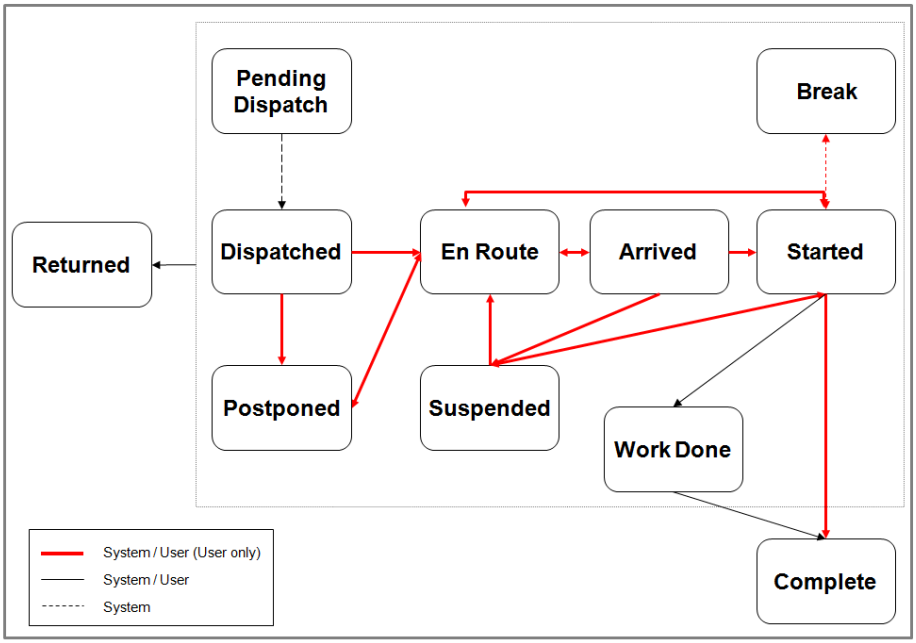
Note: The lifecycles of activities and assignments are fully described and depicted in the business object records. Refer to the **Lifecycle** tab of the **Business Object** portal for complete state descriptions; refer to the **Summary** tab for a visual representation of the lifecycles.

The following diagrams illustrate the lifecycles of each object and how they affect each other. (Refer to the maintenance portals for assignment and activity business objects for complete descriptions of the lifecycle states.)

Activity states are depicted in the following diagram:



Assignment states are depicted in the following diagram:



Incomplete State

All activity types have an Incomplete state as part of their lifecycle. This state provides a way for the crew to communicate to the scheduler that the work could not be completed for reasons beyond its control. For example, if the crew attempts to access a building but the building is not at the specified location, the crew can mark the assignment as incomplete. This bypasses any completion rules and allows the crew to move on to the rest of its work.

Work Done State

All activity types also have an Work Done state as part of their lifecycle which can be used by crews that are not connected to the server, such as SMS crews, to mark their work as done and move on without sending either a complete or incomplete status to the server. This state queues the work so that the crew can enter the required completion information later when they are logged on.

What is a Complex Activity?

Activities that require multiple shifts to complete are referred to as complex activities and are typical of, but not limited to, construction work.

The scheduler splits the complex activity into several crew visits to the site and progress is tracked per visit, similar to a regular activity. An assignment is created as usual for a visit for the crew to provide status updates and completion information. Refer to online help and user documentation to learn more about complex activities.

Consider the following when designing your complex activity types

- Complex activity type sets the average and minimum site time per visit. The scheduler splits the long complex activity into visits of average length when possible but no less than the minimum requirement which may be used, for example, to allow the crew some time for setup and tear down.
- Work can be limited to the same crew, or multiple crews may be allowed. This is defined on the complex activity type but may be overridden on the complex activity itself. If the Same Crew value is set to "Yes" one crew should work the job from start to finish. i.e., all visits for the complex activity would be planned to be performed by one crew.
- Visits are scheduled sequentially, one visit per shift, unless a user manually allocates them differently. Visits can overlap for a certain percentage of the time defined as a scheduler configuration parameter (for example to allow time for one crew to pass on or debrief another crew that is taking over the activity).
- When a crew completes a visit, the crew or dispatcher can indicate whether the work is complete or adjust the remaining time if it is not complete. As a result, the scheduler re-optimizes the whole complex activity, and may add or subtract future visits and/or adjust their length. If more time is needed to complete the activity, the scheduler simply creates another visit or appends to an existing visit. You can configure the activity type to not allow duration adjustments at all or only warn the user if they have exceeded a specific percentage.
- By default the host system is updated upon completion each visit assignment and when the entire complex activity project is complete. You may configure the algorithm on the assignment business object completed state to not send a message to host on assignment completion.

As an example, a complex activity might be set up that is 20 hours in length to be scheduled over 4 work days with the minimum visit time set as 2 hours. The scheduler would split the activity into 4 visits (visit 1 is 6 hours, visit 2 is 6 hours, visit 3 is 4 hours, and visit 4 is 4 hours).

Complex Activity and Visit Lifecycles

The lifecycle of a complex activity is very similar to a regular activity in that it is validated and is sent to the scheduler to be scheduled and it can be held, canceled and completed. It differs in that

as a long task it is never queued for dispatched and is not associated with a current assignment. Instead a complex activity is associated with visit tasks and waits passively for the work to be completed over time as crews complete their visit assignments.

A complex activity has to be explicitly marked complete by the last crew that completed the entire work. If the complex activity has expired and no crew stated work is complete the system handles the situation as per activity expiration rules on the activity type. A complex activity can be completed by a crew, by a user online or by the host system.

A visit is created only by the scheduler. In the same there is no meaning to canceling a visit by a user or a process while the complex activity is still being scheduled as the scheduler would just created more to handle the work. The system automatically cancels visits when they are no longer needed.

A visit's lifecycle is very similar to an activity in that it is queued for dispatched, manages a current assignment and can be canceled and completed. It differs in that it is merely a tool to split a long task to manageable smaller tasks and as such does not capture any business information nor activity type specific business rules. Business information and rules are captured on the complex activity, not the visit, and it is the complex activity business object reference the corresponding assignment to create for it.

> **Warning!** There is a single visit business object to handle any kind of activity and it should never be extended.

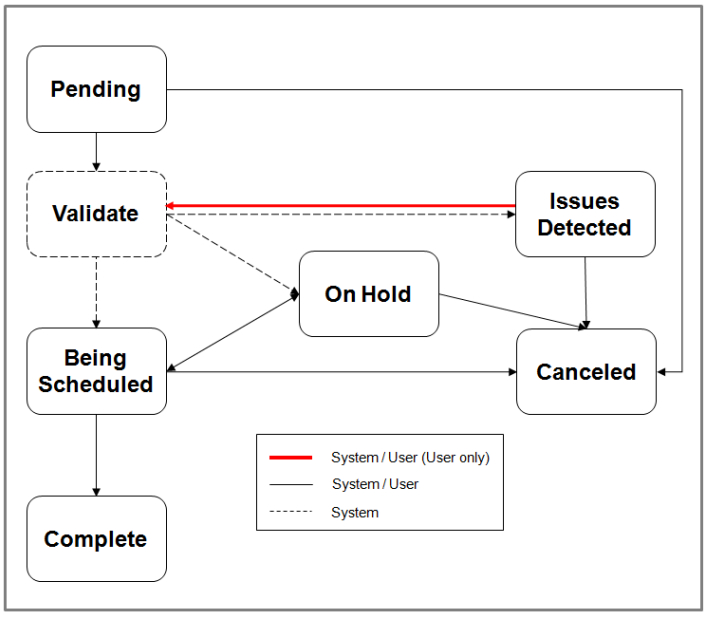
Same assignment business object, structure and rules are used for a regular activity's assignment and a visit's assignment. The lifecycle of an assignment affects the lifecycle of its visit and complex activity as follows

- When an assignment is completed, its visit is also completed.
- If the crew did not work the assignment at all the visit is rescheduled.
- If the crew stated work is complete then the complex activity is also marked complete.
- If the crew provides an adjusted overall remaining duration the complex activity is updated accordingly. This may result in more or less visits created and canceled by the scheduler.

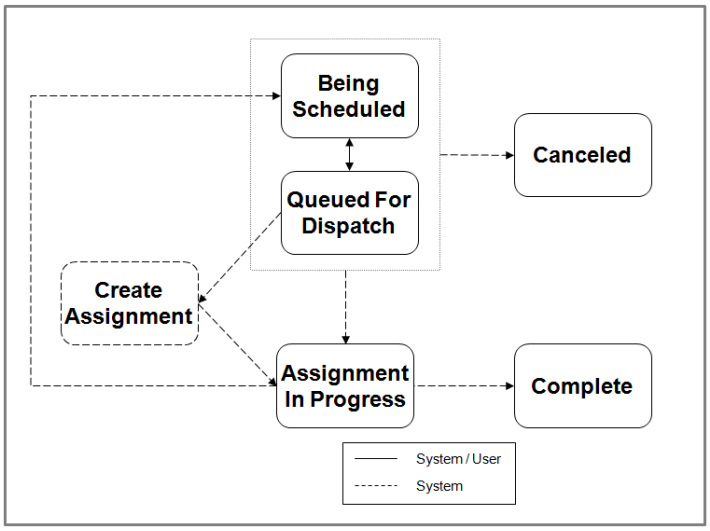
  Note: The lifecycles of complex activities and visits are fully described and depicted in the business object records. Refer to the **Lifecycle** tab of the **Business Object** portal for complete state descriptions; refer to the **Summary** tab for a visual representation of the lifecycles.

The following diagrams illustrate the lifecycles of each object and how they affect each other. (Refer to the maintenance portals for assignment and activity business objects for complete descriptions of the lifecycle states.)

Complex Activity states are depicted in the following diagram:



Visit states are depicted in the following diagram:



Activity and Assignment Business Objects Control Everything

The lifecycle, hierarchy, options and algorithms defined for the activity and assignment business objects control what activity information is received from and sent back to the host, what data is sent to and received back from the crew, what dispatcher actions are supported and what happens when each action is invoked, what crew actions are supported and what happens when each is invoked, and virtually every aspect of activity and assignment processing.

If needed, create a new activity business object. It is best to create it as a child or sub class of the activity business object provided with the base package so that the new business object will contain all the core business rules and behavior of the parent object. This will save a considerable amount of time and effort.

Refer to Appendix B: Algorithm Entities for more information about activity and assignment related algorithms.

Designing Activity Types

The base product can be used by many different types of businesses and can support a diverse set of activities ranging from field service orders, to installations and inspections, and even home deliveries. The underlying processes of scheduling, routing, and data transfer are quite similar; the primary differences are in the details, such as the implementation-specific data to be captured and how that data should be validated and processed.

Activity types are unique in that they typically require some amount of customization to adequately represent the data and business rules associated with a particular type of work. The base system provides a standard activity business object that can serve as the foundation for all custom activity types. This business object provides the basic logic to manage the activity lifecycle, handle scheduling and routing, and transfer data between the host and the mobile devices.

On top of this strong foundation, each implementation can build custom business objects by extending what is in the base system, rather than creating them from scratch. The customization process should begin with an analysis of what already exists in the base system, followed by a thoughtful design phase that identifies exactly what additional information and processes are needed. Only when the design phase is complete should an implementer begin using system configuration tools to create or customize system entities.

Below is a high-level summary of the steps required to create custom activity types for your organization.

1. Review the activity business object provided with the base package to familiarize yourself with its logic and related entities. Note that the base product's assignment business object is referenced as an option on the activity's business object. Review it as well.
2. Review and optionally customize activity-related algorithms. Refer to Appendix B: Algorithm Entities for more information.
3. Identify all custom activity types required for your implementation.

   Note: You will typically need to create a different activity type for each type of work that requires different details to be sent from the host, completion details to be sent to the host, or validation logic. For example, one activity type might require that mobile workers be sent a payment history as part of the activity detail, while another might send a prior service record. Similarly, one activity type might require that mobile workers capture a particular kind of measurement or device reading, while another might capture customer satisfaction ratings; both types would require custom logic to validate the captured data.
4. The base package provides the To Do Type, M1-TSKTD, to capture activity data issues. Review its details. If you want the system to generate To Dos, associate it with To Do roles as per your business needs. Refer to the Oracle Utilities Application Framework documentation for more about To Do Types and Roles.
5. For each activity type you identified above, do the following:
   - Identify all data to be received from and sent back to the host.
   - Design the UI maps to be used to display and capture data. You will need to design screens for the server application and, if MDTs are used in the fields, the mobile application as well. Remember that you can clone the screens associated with the base business objects and then modify them as needed.
   - Identify all behavior needed to extend the base business object for this activity type, including algorithms that will be needed to handle validation of any data to be captured, monitor processes, etc. Make a list of these items so you can be sure to address all of them in the development process.
   - Identify the data that needs to be synchronized to the crew. Do not include any data the crew does not need to know in order to perform their work.

Defining Your Activity Types

The following describes the high-level steps needed to define a new activity type:

1. Define an activity business object that represents all data and behavior required for activities of that type.

   The activity business object should be defined as a child of the base activity business object; thus, all events associated with the parent business object are automatically supported for its child business object.

   The business object's schema should include the base activity business object schema as well as any incoming details sent from the host, but not completion details.

   Finally, associate the UI Maps, custom algorithms, and service scripts required to display or capture data, perform validation, or handle any other custom business processes.

2. Define a data area for the dispatch details that need to be synchronized to the crew, as identified during the design phase. The data area should include the base dispatch details data area associated with the base assignment business object, as well as specific details relevant to this type of activity. The **Task Synchronization** process uses this data area to dynamically send only this subset of details to the crew.

3. Define an assignment business object that represents all data and behavior required for an assignment of this activity type.

   The assignment business object should be defined as a child of the base assignment business object.

   Its schema should include the base assignment business object schema as well as completion details relevant to this type of activity to be sent to the host.

   As with the activity BO, associate the UI maps, custom algorithms, service scripts, etc., required to display or capture data, perform validation, or handle any other custom business processes.

   Reference the dispatch details data area described in the previous step as a business object option on your assignment business object.

4. Reference the assignment business object created in the previous step as a business object option on your activity business object.

5. Define an activity type record (using the **Task Type** portal, accessible from the **Admin** menu). When prompted, select the appropriate activity business object defined earlier.

   For each activity type record, you will need to provide the following information. Consider each item carefully before you create the record.

   - The host system for this activity type
   - How long an activity of this type typically takes to complete
   - Whether or not this type of activity should be automatically dispatched
   - Whether or not an acknowledgement from the crew should be required for this type of activity (emergency activities typically require acknowledgement)
   - The priority queue, scheduling priority information and priority profile (priority profile defines attributes such as sequence, whether the activity type is mandatory or optional, and whether or not an offset is use. Note that emergency activities must also be set as mandatory.)
   - The appointment booking group to use
   - The work profile to use
   - Any scheduling cost factors

- Whether or not activities of this type need to be allocated to the assigned crew at some predefined time prior to shift start
- How expired activities of this type be handled. The activity business object you created earlier for this activity type
- Skills and equipment required to complete activities of this type
- Equipment required to complete activities of this type
- If your organization uses contractors decide whether this type of work is eligible for contracting. Refer to Contractor Management Basics for more information.
- Review complex activity configuration such as duration limits as well as same crew requirement.
- Completion remarks associated with activities of this type
- To Do setup information

Activity-Related Scheduling Cost Factors

The following scheduling cost factors are defined at the activity-type level:

- Scheduling Priority (Relative Shift Promotion Cost)
- Preferred Time Factor (Preferred Time Factor)

The scheduler uses the following activity-based parameters when assigning tasks to crew shifts and optimizing crew shift schedules:

- Scheduling Priority (Relative Shift Promotion Cost)
- Relative Late Cost
- Relative Window Cost
- SLA Flexibility

POU Task Types

If the resource type associated with a POU is set to crew, the system will automatically generate POU tasks for all crews in the specified service areas and incorporate them into the crew's shift schedule. A POU task type record defines the POU task business object to use when creating POU tasks. The system uses a single type of POU task, which is defined on the Master Configuration, to generate all POU tasks.

Service Classes

Service class is a broad categorization of activity types which can be used to control work assigned to crew shifts by the scheduler. Refer to Controlling Work Assignment for more information.

If a service class represents a primary function it might also specify procedure types a crew has to complete before they take on that function. Refer to Procedure Types and Procedures Basics for more information

Service classes may also be used to restrict the crews and activities monitored by a dispatcher. Refer to Dispatch Areas for more information.

If your company uses external contractors your company may offer them a certain capacity of work in a specific service area and of a specific class of work. Refer to Contractor Management Basics for more information.

Review your activity types and group them to various service categories to meet your business requirements. One activity type can belong to multiple service classes.

Dispatch Areas

A dispatch area is a predefined set of service areas and service classes that can be monitored by one or more dispatchers. A dispatcher record identifies one or more dispatcher areas the dispatcher is allowed to monitor. When a dispatcher logs on to start a shift, they can select which of the authorized dispatch areas they will monitor during that shift. The system determines which activities and crews the dispatcher is responsible for monitoring during their shift based on the selected dispatch areas:

- A crew's shift matches a dispatch area if at least one of the shift's service areas and at least one of its allowed service classes are included in the dispatch area definition.
- An activity matches a dispatch area if its service area and at least one of its service classes are included in the dispatch area definition.
- It is enough for an activity or shift to match one of the monitored dispatcher areas to be monitored by the dispatcher.
- A single dispatch area may be monitored by multiple dispatchers.

Before you set up your dispatch areas, study the distinct combinations of service areas and service classes to be monitored by dispatchers.

Alerts

This section describes key concepts and configuration information related to alerts.

What is an Alert?

An alert is a situation in which predefined conditions have been met that require user attention or intervention. An alert could be raised for a crew's shift or a task.

The system automatically assigns an alert to a single logged-on dispatcher based on configurable business rules. Once assigned, the alert appears on the user's **Alert Queue** dashboard zone.

You can configure alerts to be recycled if the assigned user doesn't acknowledge the alert within a configurable amount of time. Alerts assigned to a user are always recycled when the user logs off.

You can configure alerts to snooze for a configurable amount of time once the user acknowledges the alert. The alert is dropped from the user's alert queue while the user tries to address the issue, and will reappear after that amount of time if the conditions are still true for the alert.

To make sure that the underlying condition(s) of an alert are resolved properly, you can design business rules to constantly evaluate and finally close the alert automatically.

An alert can also be configured to expire, i.e., be closed, after a certain amount of time.

Alert Type Controls Everything

Alert type defines attributes common to all alerts of that type. Alert type attributes include the creation and evaluation algorithms, the threshold for creating alerts of this type, whether or not alerts of this type can be manually closed, and the length of time before the alert expires, is recycled, or is allowed to snooze.

Alert type also defines the business object used to create and control the structure and behavior of alerts of this type.

The base product provides a set of business objects and algorithms that can be used to create and evaluate different types of alerts. The base product does not provide the actual alert type records.

At a bare minimum, an implementation needs to create alert type records that reference the creation and evaluation algorithms and the alert business objects provided in the base for each alert type. An implementation can also create additional entities to support other types of alerts.

> **Note**: Some alerts, like the missed appointment alert, do not have evaluation algorithms because the condition that causes the alert to be generated is already in the past. There is no condition to research or resolve; the alert is simply a notification.
>
> Likewise, some alerts do not have creation algorithms because they are generated when a system event, such as a synchronization timeout, occurs. The activity cancellation alert is another example; it is raised whenever the host requests to cancel an activity that is already dispatched.

Audible Alerts

You can configure alerts to make a sound to help get the attention of the dispatcher when they appear on the alert queue. Specify the sound files to use with Master Configuration sound play parameters.

Designing Alert Types

The following is a high-level summary of the steps involved in designing alert types for your organization.

1. **Understand the business conditions for which to raise alerts of this type.** If the alerts are to be raised upon a specific event, design an algorithm that will be triggered by the event to create an alert. Many alerts are not event-driven, but rather are raised based on a situation that is determined by frequent analysis of the data. For such alerts, an Alert Creation algorithm needs to be developed based on the business rules. The Alert Creation algorithm is called by the monitor plug-in associated with the alert type business object. For more information, refer to the alert batch controls section in Appendix C: Batch Controlsand to Appendix B: Algorithm Entities.

   Most **Alert Creation** algorithms require some kind of threshold. For example, you might want to raise a crew idle alert when a crew has been idle for more than a specific length of time. The idle time, in minutes, may be defined on the alert type.

2. **Determine whether alerts of this type should automatically expire after some time.** The expiration time, in minutes, may be defined on the alert type. If not specified, alerts of this type do not expire.

   An alert should only be allowed to expire if it is simply a notification and does not require a dispatcher to take any specific action.

3. **Determine whether the system should automatically recycle an alert of this type if it is not acknowledged in a timely manner.** The recycle time, in minutes, may be defined on the alert type. If not specified, alerts of this type are never recycled based on time; however, they will still be recycled when the assigned dispatcher logs off.

4. **Determine whether an alert of this type should snooze for a configurable amount of time after a dispatcher acknowledges it.** As a general rule, you should allow an alert to snooze if it is important that the alert be closed only when the underlying issue has been resolved, and you want to allow the dispatcher for more time to resolve it. When the snooze time expires, the system evaluates the underlying condition and closes the alert if the condition no longer exists. The snooze time, in minutes, is defined on the alert type.

   You can also decide which alert types should allow the dispatcher to adjust the snooze time. If not specified, an alert of this type automatically closes once a dispatcher acknowledges it.

5. **Understand the business rules that allow the system to automatically close an alert of this type.** When applicable, design a corresponding Alert Evaluation algorithm and specify it

on the alert type. The Alert Evaluation algorithm is called by the monitor plug-in associated with the alert business object. For more information, refer to the alert batch controls section in Appendix C: Batch Controlsand to Appendix B: Algorithm Entities.

6. **Determine whether to allow dispatchers to manually close alerts of this type even if there is an evaluation algorithm.** There may be situations where an alert cannot be automatically closed by the evaluation algorithm. For example, if a crew's mobile device loses its connection for a period of time and the crew communicates via phone to the dispatcher, you might want to allow the dispatcher to close the alert so it will be removed from the queue.

   Note: It is a best practice to let the system evaluate and close all alerts. If you have to allow manual closure, please be sure to build this logic into the Alert Creation algorithm to avoid raising repeated alerts for same issue.

7. **Determine the priority of alerts of this type relative to other alerts**. The priority dictates the order by which alerts are presented to dispatchers. The priority may be defined on the alert type.

8. **Review the alert business objects provided with the base package to familiarize yourself with their logic and related entities**. Choose the one that's most appropriate for your alert type. If needed, create a new alert business object with a different lifecycle and business rules to meet your organization's business requirements.

9. Define the alert type and all its attributes, including the alert business object and calculation algorithm.

10. Determine if the alert requires a sound to be played.

Alert Batch Controls

Alert creation and evaluation algorithms are called by a batch process.

- The **Alert Type Monitor Batch Control** is responsible for creating alerts. It is critical that this process runs frequently enough to raise alerts in a timely manner, so the underlying condition can be addressed promptly.
- The **Alert Monitor Batch Control** is responsible for evaluating alerts. It is important that this process be called frequently so that alerts can be closed promptly when the underlying condition is resolved.

Alert Queue

The **Alert Queue** is a dashboard zone where users can view and manage alerts. The **Alert Queue** zone should be enabled in User Preferences for all users who are responsible for monitoring alerts, and the auto-refresh settings should be set to refresh frequently.

Dispatchers can also identify a backup dispatcher in the **Backup Alert Queue** zone. This zone lists alerts of other dispatchers that the shift dispatcher is backing up. If used, this zone must be enabled just like the alert queue zone.

KPIs

This section describes key concepts and configuration information related to key performance indicators (KPIs).

What is a KPI?

A key performance indicator (KPI) is a quantifiable measurement that supports dispatcher decision making and in-day exception handling. For example, a KPI might measure the number of out-of-service crews or the number of appointments in jeopardy of being missed.

KPI Record Controls Everything

A KPI record defines what is being measured and how to measure it. It specifies the calculation algorithm to use and the threshold values used by that algorithm.

Some KPIs measure the same underlying conditions as the alerts raised by your organization. For those KPIs, the base product provides a KPI calculation algorithm that counts the number of alerts whose related activity or crew shift is in the user's monitor data scope.

For KPIs not related to alerts, a new plug-in must be created. For example, if you want to create a KPI that counts the number of emergency activities received or the number of breaks canceled, create an algorithm to perform the calculation then create a KPI record that references the calculation algorithm.

The KPI calculation plug-in spot supports two modes, Summary or List. If called in the Summary mode, the algorithm returns the summary value of the KPI. If called in List mode, it returns the detail of the KPI.

The base system does not provide any KPI records. An implementation is responsible for creating all KPI records and then associating them to Dispatcher records, indicating that the dispatcher can monitor the KPI on their **CDI** portal.

Designing Your KPIs

The following is a high-level summary of the steps involved in designing KPIs for your organization:

1. **Determine what the KPI will measure.** Identify whether the KPI is related to activities or to crews.
2. **Determine whether there is an existing alert type that measures the underlying condition.** If so, you can use the KPI calculation algorithm provided with the base package. If not, you must create an algorithm to calculate the KPI. The calculation algorithm may be defined on the KPI record.
3. **Identify the beginning of the normal, warning, and severe ranges for the KPI.** These values may be defined on the KPI record.
4. **Review the KPI business object provided with the base package to familiarize yourself with its logic and related entities.** If needed, create a new KPI business object to meet your organization's business requirements.

Chapter 5

Attachments

Attachment functionality supports both upload and download of attachments with all file types on laptops. This supports mobile workers in sending and receiving additional information regarding activities such as pictures of an activity site, detailed reports completed in Word documents, recorded sound or video, etc.

To configure attachment functionality:

- Each type of attachment must be defined on the **MDT Type** as a supported capability by devices of that type.
  Note: It is your organization's responsibility to have the appropriate tool or application to view the attachment on the mobile device.
- Review **MDT Types** to determine the max size for each device being used in the field and use the **Attachment Storage Size** field to set the appropriate value.
- Several standard attachment file types are delivered out of the box, such as PDF, Word, Excel, or other document types. However you can add additional file types by creating additional business objects that subclass the **F1-ATCHMT** maintenance object with the same schema as the out of the box attachment file type business objects.

  Note: Search for the **F1-ATCHMT** maintenance object to see the full list of out of the box attachment types.
- Consider where attachments must be automatically pushed to mobile workers and add these as deployment parts.

Process Notes

Dispatchers and mobile workers will need to be trained on your business practices related to when and how to capture and upload attachments for activities. Workers will also need to learn how to download attachments that are included with activities. If attachments are set so that they are not automatically sent to the mobile device, workers will need to learn how to obtain these attachments when they need them.

Limitations

- When considering limitations for attachments, it is important to make a distinction between uploading and downloading attachments.
  Upload: Upload functionality for all file types is only available for laptops. For Windows Mobile and Android mobile devices, pictures or audio files (considered “captures”) can be attached and viewed, but not other types of files.
  Download: All attachments can be downloaded on all devices, however if the file type is not supported on the device, it cannot be opened or viewed.

Not supported for browser based mcp MCP. This applies for captures and attachments.

Chapter 6

CDI Configuration

This section describes topics related to configuration of the **Common Dispatching Interface** (CDI) portal and related functions, including the following topics:

What is the CDI?

The **Common Dispatching Interface** (CDI) portal is the primary point of user interaction for management of field resources and tasks. It provides a visual representation of scheduled shifts and their assigned activities, and provides access to all dispatcher functions.

CDI is for Dispatchers Only

Only dispatchers can access the **CDI** portal. When a dispatcher attempts to access the **CDI** portal, the system automatically displays the Logon screen if no active dispatcher shift already exists. Dispatcher shifts are not created until the dispatcher logs on to start a shift.

To ensure that the CDI is working properly and users can access its full functionality, please verify the following:

- The user has been set up as a dispatcher in the system.
- Dispatch areas have been defined, and the dispatcher record identifies the dispatch areas and KPIs the dispatcher is responsible for monitoring.
- The **Alert Queue** and the **Backup Alert Queue** zones are properly set up, as described in the **Alert Configuration** section.

  Note: Before a dispatcher can view the **CDI** portal, he or she must log on to start a dispatcher shift. When a dispatcher attempts to access the **CDI** portal, the system automatically displays the Logon screen if no active shift already exists.

The Scheduling Gantt

The system uses a **Gantt** zone to display a graphical representation of the crew shift schedules being monitored by the logged-on dispatcher.

The CDI also supports a geographical map view of the dispatcher's monitored area. The map is launched from a toolbar button on the **CDI Scheduling Gantt** zone.

Refer to Chapter 8: Specialty User Interface Configuration for more information on how these are configured.

Frequent Data Refresh

Key operational zones should be configured to automatically refresh their data to reflect latest updates.

Ensure the automatic refresh interval is properly set up for the appropriate zones. At a minimum, Oracle recommends setting the CDI-related **KPI Summary** and **Gantt** zones and the **Alert Queue** dashboard zone to automatically refresh frequently. The refresh interval for these zones is defined on their maintenance portals. Individual users can override these settings in the zone refresh parameter in user preferences.

The automatic refresh interval for the CDI Map is defined as a multiple of the refresh interval of the Gantt. For example, if the Gantt's automatic refresh interval is 60 seconds and the multiple is defined as 5, the map will be refreshed every 5 minutes. That multiple is defined as a parameter of the **Gantt** zone.

Dispatching Functions

Dispatchers resolve day-to-day exceptions by performing common dispatching functions. These functions can be performed from the **Gantt** or **Activity Search** zone on the **CDI** portal or from the activity maintenance portal.

These functions are implemented in dedicated algorithms associated with the activity business object. Review the algorithms that control dispatching functions to ensure that they will meet your organization's needs:

- Allocate to Shift
- Cancel Activity
- Defer Activity
- Lock Depot Run(if depot functionality is used)
- Unassign

Refer to Appendix B: Algorithm Entities for more information about these algorithms.

Chapter 7

Procedure Types and Procedures

This section provides information about procedures and procedure types including:

Procedure Types and Procedures Basics

The system can be configured to include procedural instructions or questionnaires to be completed by mobile workers prior to taking on a specific primary function when starting a shift or changing their function mid shift, using a vehicle or starting work on an activity. If the procedures fails the corresponding business action cannot commence.

Usually procedures relate to safety precautions or regulatory requirements. For example, the mobile worker may need to complete a vehicle inspection at the beginning of a shift, or review safety instructions for a hazardous task.Procedure enforcement is applicable to the mobile application only and is embedded as part of the following process flows

- Request to start a shift
- Request to change a started shift's primary function
- Request to change a started shift's vehicles.
- Every time the crew starts an activity of a specific type. If they suspended and start it later again they would be requested to complete the procedures again

Procedures reference the entity they are related to which can either be a shift, a resource or an activity. Procedures related to primary function reference the shift, those related to utilizing a vehicle reference the vehicle they were completed for and those related to starting an activity reference the activity.

As procedures are created, processed and completed, the system creates a record in the **Procedure** portal to track the information, attendance (if applicable), and results. Procedures cannot be manually created in this portal. The **Procedure** zone on activities shows a listing of which procedures are required for the activity and which procedures were completed.

The following sections describe how to configure the system to enforce your business procedures as part of these events.

Procedure Definition

Procedure types control a particular type of procedure. They mainly control which business object is used when creating a new procedure of that type, thus dictating the governing form presented to the crew as well as the business rules involved in completing it. The base product supports a configurable questionnaire type of procedure but you can implement more sophisticated procedure forms using customer procedure business objects.

The procedure type establishes whether the procedure is applicable to an activity type, a vehicle type or for a primary function. After creating the procedure type, link it to the appropriate definition.

This also establishes a valid effective date and an expiration date for the procedure as needed. Base package options also include the ability to set whether or not the procedure can be overridden and/or whether or not taking attendance is required.

Your business practices will determine how the mobile worker should proceed in cases where the procedure fails, whether or not procedures can be over-ridden, and whether or not the mobile workers are required to take attendance for the completion of procedures.

Shift Based Procedures

Procedure types specified on a **service class** indicate procedures that must be completed when the crew starts a shift referencing this service class as their primary function or each time they request to change their started shift to assume that function.

Procedure types specified on a **vehicle type** indicate procedures required to be completed before the crew is allowed to use a vehicle of this type. A separate procedure is required for each vehicle of the specific type each time such vehicle is used.

At shift start time if any procedures are defined for the crew's primary function or related to any of the vehicles they are about to use they are sent to the crew's mobile device, the crew has to complete them and they have to all pass before the crew can start their shift. In the same way, if a crew changes their primary function or vehicle allocation mid shift corresponding procedures have to be completed before the changes are in effect.

It is important to note that the list of procedures is determined and created on the server as part of processing the crew's synchronous request and sent back to the device as part of the response to that request.

Once the list of procedures is copied to the device they are presented to the crew which then completes then one after the other. Each procedure is presented and validated as per standard business object functionality on the mobile device.

A failed procedure may or may not be allowed to be manually overridden as passed based on your procedure type configuration. Once all procedures are passed or overridden the system proceeds to perform the requested change.

Activity Based Procedures

Procedure types entered on an **activity type** indicate procedures required before the crew can start work on the activity. These procedures are based on the specific activity assigned to the shift and must be completed and pass before the crew starts the activity.

It is important to note that the list of procedures is determined and created on the server and sent to the mobile device when the activity is dispatched.

This is also true for an activity created in the field. When the activity is sent to the server it goes through the same process to handle procedures. The **Activity Type** can be set to explicitly define when safety procedure clearance is required. If the crew decided to work the activity and the activity type requires safety procedure clearance, the crew is required to be connected to receive the procedures and fill them prior to starting the work.

Determine Procedure Algorithms

Your implementation may have different rules to determine which procedures are applicable for an event.

The logic to determine shift based procedures resides in the **Determine Shift Procedures** plug-in spot defined on the shift business object.

The algorithm may be called in **Determine** mode where it only returns the list of procedures to be created but does not actually creates them. This is used to display planned procedures on the crew shift portal. It can also be called in a **Create** mode to determine and create the procedures. This plug-in is called in this mode when the shift is about to start as well as each time either primary function or resource allocation is about to be changed. Refer to Generate POUs in Appendix B: Algorithm Entities for more information.

The logic to determine activity based procedures resides in the **Determine Task Procedures** plug-in spot defined on the activity business object.

This algorithm is called when the assignment is created for an MDT crew. This also supports **Determine** and **Create** modes as described above.

> Note: Determining and creating procedures has to occur on the server so as to allow these algorithms to consult other entities and apply custom rules. This means the crew has to be connected to obtain them.

Training

Mobile workers will require training on how to process the procedures before starting work. For example, how to complete procedures in general, how to enter answers on the mobile device, what to do in cases where the procedure fail, and so on. The details will depend entirely on your business practices and how you choose to implement this functionality.

Configuring Procedures

Complete the following configurations to set up procedures for use in your implementation.

- Analyze your procedures and procedure forms to determine how they can be transformed to be used with this feature.
- Plan your procedure types by determining how each procedure type will be used. This includes:
  - **Related Entity** - whether procedures of that type are for activity types, vehicle types or service classes.
  - **Override settings** - allow or prevent a shift from starting if the questions are not answered correctly or if the procedure otherwise fails.

    Note: Make sure that the mobile workers can pass the procedure steps and/or there are easy to follow and well documented processes in place for when they do not pass or need to be overridden. For example, you may want to enforce a requirement that the worker must enter comments if they override a procedure.
  - **Attendance settings** - does the crew need to take attendance (keep a record of who was present during the completion of the steps).
  - **Related steps** - the actual steps to complete the procedure.

    The base product's procedure type and procedure business objects implement a simple questionnaire form. If your implementation uses the base procedure business objects, the **Procedure Steps** section on the **Procedure Type** indicates the specific procedure or steps that need to be followed to complete the procedure. Using

procedure based message categories, such as vehicle inspection message category or activity safety message category, you can create the questions and answers that will make up the procedure steps. These can be yes or no questions, text, numbers in a valid range, etc.

Create message categories to define procedure steps.

- All custom message categories must be numbered 90000 and above.
- It is recommended that you create a new, dedicated message category for procedures.
- Message categories must be included in the deployment type so that they are included on the deployment that is sent to the mobile workers (part of the activity deployment).

- Create more complex procedure forms as needed by implement each as a custom procedure business object using specific UI maps and script.
  - Use the **Procedure Type** portal to define your procedure types referencing the procedure business object base or custom.
  - Review existing and/or create new activity types, vehicle types, and service classes to associate procedure requirements as needed.
  - When creating procedure steps, refer to the generic procedure type business object, **M1- ProcedureType**.

Chapter 8

Specialty User Interface Configuration

This section describes special types of user interface configurations, including the following topics:

- Gantt Zones
- CDI Map
- Calendar Zones

Gantt Zones

The system uses **Gantt** zones to display a graphical representation of crew shift schedules over time. This type of zone is used on the **CDI** portal, where it displays a graphical representation of the crew shift schedules being monitored by the logged-on dispatcher. This zone type is also used on the **Crew Shift Schedule** tab portal, where it displays a graphical representation of a particular crew shift's schedule.

When a user right-clicks on a shift or task displayed in the **Gantt** zone, the system displays detailed information about the object at the top of the context menu. The logic used to display detailed information about a shift or task resides in the following plug-ins:

- **Task Detailed Information**
- **Crew Shift Detailed Information**

The logic used to derive the contents for a **Gantt** zone resides in the CDI Gantt Data plug-in.

> Refer to Appendix B: Algorithm Entities for more information about these plug-ins.

The functionality provided by this zone requires that you set up a feature configuration of type "Application Context Root," which defines the main application's root location.

> Refer to Feature Configuration for more information.

Context Menus

You can add custom options to the context menu that appears when a user right-clicks an object in the **Scheduling Gantt** zone. The system supports several customizable context menus:

Context Menu	Displays when a user right-clicks...	From the...
M1-GANTT-ACTIVITY	A single activity that is not completed	Scheduling Gantt in the CDI

Context Menu	Displays when a user right-clicks...	From the...
M1-GANTT-ACTIVITY-MULTI	Multiple activities that are not completed	Scheduling Gantt in the CDI
M1-GANTT-BREAK	A break task	Scheduling Gantt in the CDI
M1-GANTT-DEPOT-TASK	A depot task	Scheduling Gantt in the CDI
M1-GANTT-DEPOT-TASK-MULTI	Multiple depot tasks	Scheduling Gantt in the CDI
M1-GANTT-POU	A POU	Scheduling Gantt in the CDI
M1-GANTT-SHIFT	A shift	Scheduling Gantt in the CDI
M1-GANTT-SHIFT-SHIFT_PORTAL	A shift	Scheduling Gantt in the Crew Shift Schedule tab.
M1-GANTT-TASK	Any other single task	Scheduling Gantt in the CDI

To add a new item, select **Menu** from the **Admin** menu and then find the appropriate content menu from the list above. Refer to the Oracle Utilities Application Framework documentation for instructions on adding menu items to the menu. Menu lines added to these context menus must invoke a BPA script to perform the work, and the BPA script must follow a specific schema. Each context menu provided with the base package contains at least one item that serves as an example of how to implement such a menu item.

The items you add will appear on the on the appropriate entity's context menu in addition to the standard menu times.

Gantt Colors

Customize the colors used to represent shift and task states by copying the base record from managed content to your own record.

1. Navigate to **Admin** -> **Managed Content**.
2. Search for and select the M1-GanttStyles record.
3. Copy to your own record.

   You must create a custom managed content record that is a copy of M1-GanttStyles then modify the copy.
4. Adjust the colors as needed using html hex codes.
5. Override the URL parameter on the **Gantt** zone to reference their custom managed context record instead of the base.

CDI Gantt Filters

The base product provides out of the box CDI Gantt Filters for a dispatcher to filter the shifts they view on the Gantt on CDI portal. They appear when you click on the filter icon on the Gantt on CDI portal. Filters are defined in the CDI Gantt Filters extendable lookup.

You can inactivate a base filter by setting the corresponding record in the extendable lookup as inactive.

You can introduce custom filters as needed to meet your business requirements. Follow the steps below to set up a new Gantt predefined filter:

- Create a data explorer query zone that returns a list of shift identifiers based on your business requirement. The zone must contain the shift id and nothing but that column as an output column.
- Create a data explorer (FWLZDEXP) Business Service for your zone.
- The Business Service must map the zone's returned Shift ID column to an element named 'shiftId'
- If you wish to default any input parameters, you may do so by using the standard schema standards (e.g. default = "xyz")

  Warning! It is your responsibility to ensure the SQL is efficient and references the dispatcher's data scope. The Gantt algorithm does not validate that the returned shifts are indeed relevant to the dispatcher. Refer to the base filters for examples.
- Add your business service to the CDI Gantt Filters extendable lookup.

Gantt Defaults

You may override the default Gantt display layout by creating a customized Gantt Display Managed Content and using it to override the base record on the appropriate zone configuration.

Follow these steps to define the Gantt display layout for your implementation

- Duplicate the base “Gantt Display Defaults” Managed Content object
- Customize the new Managed Content object as desired:
  - To restrict the list of available display columns, remove "Column Name" entries from the Display Columns group as needed
  - To modify the 'default' Gantt layout (that is, the view of the Gantt a user will see if they have not set their own default) mark specific columns as default.
  - You may also modify the following attributes of each column:
    - The sequence (order) each column appears at
    - The default width of each column
    - A default sort (ascending, descending)
    - Sort sequence (if sorting by more than one column)

Add an override to the CDI Gantt Zone's "ADF Application URL" to refer to the new managed content object (by replacing the 'gantt display managed content' parameter)

CDI Map

The **Common Dispatcher Interface** (CDI) supports a map view of the dispatcher's monitored area. The **CDI** map is launched from a toolbar button on the **CDI Scheduling Gantt** zone.

The CDI Map functionality requires that you set up a feature configuration of type “Map Server Configuration.”

Refer to Feature Configuration for more information.

Map layers are controlled by the “CDI Map Layer” extendible lookup.

Each map layer definition indicates whether the layer should be loaded and displayed when the CDI Map is initially loaded. You can override the default indication as needed.

You can add to the extendable lookup custom layers of the following types:

- A custom layer may reference a map viewer theme name you have predefined in the map viewer repository. Note that this theme must not assume any context, i.e. it is called without any input from CDI map.
- The system allows you to define geographic areas for various usages. A custom layer may be added to show all active geographic areas of a specific usage. Refer to the user guide for more information on how to set up geographic areas.

When a user hovers on a point on the map that is related to a shift or task, the system displays detailed information about the object in a context window. The logic used to display detailed information about a related shift or task resides in the following plug-ins:

- Task Detailed Information
- Crew Shift Detailed Information

Refer to Appendix B: Algorithm Entities for more information about these plug-ins.

The system calls the service script configured on the map layer entry in the extendable lookup to format the information displayed when the user hovers over an icon in that layer. You can use a custom service script as needed so long as it include the base script schema as part of its schema to ensure upgradability.

Calendar Zones

The system uses **Calendar** zones to display events or items occupying time in a calendar format. For example, the **Calendar** zone on the **Crew** portal displays a crew's calendar of shifts and meetings.

Logic to derive the contents for a **Calendar** zone resides in a plug-in. Refer to Appendix B: Algorithm Entities for more information about this plug-in.

The functionality provided by this zone requires that you set up a feature configuration of type "Application Context Root," which defines the main application's root location.

Refer to Feature Configuration for more information.

Chapter 9

Mail Management

This section describes how to configure mail management functionality, including:

Mail Management Basics

Mailing functionality is configured using a number of business objects. These dictate how mail is sent, how it is received, and who receives which messages.

When a mail message is created its business object determines the list of recipients it should be sent to and creates a recipient mail message for each recipient. The business object on the recipient mail record is responsible for delivering the message also to the recipient mobile device when applicable. Review the base package business objects to better understand how mail is processed and to customize these rules as needed.

At any time a user can view mail they sent or received via the **My Mail** portal.

Mail Management User Interface

Users are notified of new messages and have access to opening mail via the **My Mail** portal and the **Mail Summary** dashboard zone.

My Mail Portal

The **My Mail** portal provides access to view mail you have received or sent. Any user with a valid login can use this portal to see their messages. This is also useful for non-MDT or SMS crew members who do not have access to mail from a portable device. You can create new mail, reply, acknowledge mail you received and archive your mail using this portal.

Mail Summary Dashboard Zone

You can also manage your mail using the **Mail Summary Dashboard** zone. Make sure that users who should have access have this zone added in their preferences.

Mailing Lists

Mailing lists define a list of users for mail distribution purposes. When mail is sent to the list, each user receives a copy. Set the status of mailing lists to active or inactive to affect their availability to users. Mail cannot be sent to inactive lists.

Mailing Groups

Mailing Groups differ from Mailing Lists in that a group is dynamic. For example, a mailing group changes based on the logged on or logged off status of members of the group. The logic to interpret the business meaning of each group code resides in an algorithm associated with the mail business object. Use this business object to customize your group codes.

Mail Templates

Mail message templates establish standard settings for the fast creation of mail messages. Users can then select the template when creating new mail and the pre-defined subject, message and other settings are automatically copied to the new message from the template.

You should configure appropriate mail templates to handle standard situations which may arise in your business process.

Chapter 10

Hierarchies

This section provides information about crew and service area hierarchies including:

Hierarchies Basics

Hierarchies organize crews and service areas into logical structures of up to three levels that represent their position in relationship to other crews or service areas, respectively.

Crews can be grouped by how they are managed, internal versus external, etc. Service areas can be subdivided into territories, regions, offices, etc.

Once they are configured, hierarchies appear as search options in the various queries where crews and service areas can be searched. Filtering by hierarchies can be very helpful toward organizing information and allowing dispatchers and other users to more easily locate the crews, service areas, and activities that require their attention. Hierarchies can also be used for reporting purposes.

Download a hierarchy from the info zone in the **Crew Hierarchy** or **Service Area Hierarchy** portal using standard **Export** functionality.

Configuring Hierarchies

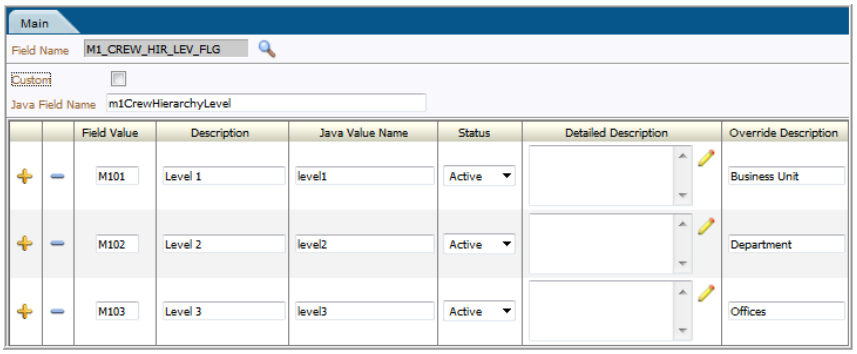
You can define hierarchies of up to three levels to establish categorizations for crews and service areas. Crews and Service Areas must all belong to the lowest level defined for their respective hierarchy. For example, if crew hierarchy uses two levels then all crews must reference a level two node in the hierarchical structure.

Enable hierarchy levels by configuring the following lookups:

- M1_CREW_HIR_LEV_FLG
- M1_SER_ARE_HIR_LEV_FLG

You can define up to 3 levels with level 1 being the highest, level 2 below level 1, and level 3 below level 2. Enable each level by populating the override description to describe its role in your business structure.

Example Levels:



Create your full hierarchy structure via manual entry or by uploading a spreadsheet:

- **Manual**: Access each of the hierarchy portals, create new, and enter details according to your business requirements.
- **Upload**: Access each of the hierarchy portals, select **Upload Hierarchy**, and upload a pre-populated spreadsheet:

  Example format:

	A	B	C	D
1	Hierarchy Node Name	Description	Level	Parent Node Name
2	CRH0	Crew Hierarchy 0 Level 1	M101	
3	CRH1	Crew Hierarchy 1 Level 1	M101	
4	CRH2	Crew Hierarchy 2 Level 2	M102	CRH1
5	CRH3	Crew Hierarchy 3 Level 2	M102	CRH1
6	CRH4	Crew Hierarchy 4 Level 3	M103	CRH2
7	CRH5	Crew Hierarchy 5 Level 3	M103	CRH2

  The spreadsheet should include the following columns:

  - **Hierarchy Node Name** - the code for the hierarchy node being created
    Note: All hierarchy nodes must be unique across the system, regardless of their hierarchy and level.
  - **Description** - description for hierarchy node being created
  - **Level** - the code for the level, as defined above.
  - **Parent Node Name** - the code of the node in the level immediately above the node being added (creates the relationships in the hierarchy).
  - (optional) **Delete Y/N** - A delete can be added to the spreadsheet for data clean up. You would not use this the first time uploading the hierarchy, but you could use it to correct information if necessary. See below for more details.

    Note: Make sure that parent nodes are entered first with children entered after them, i.e., you must create the parent before the child can be added.

It is best to set up your hierarchies first so that when you create crews and service areas you can reference the lowest level they belong to. You can always update them later as needed.

Uploading Hierarchy Corrections

You can upload a new spreadsheet with hierarchy corrections if necessary. This can be used to change information as well as to delete entire nodes. Review the following scenarios for examples and how to address the correction.

- **Problem**: Several of the node descriptions have a typo due to a copy/paste error.
  Solution: Re-upload the spreadsheet with only the descriptions modified. If the spreadsheet includes the same Hierarchy Node Name, Level, and Parent Hierarchy Node, the description information will simply be overwritten with the correction.
- **Problem**: Some of the entered hierarchy nodes are not needed.
  Solution: Add a a delete column to the spreadsheet with values set as Y for each Hierarchy Node name needing to be removed. Those nodes will be deleted upon upload. Note: Make sure that child nodes are removed first with parent nodes removed after, i.e., you must remove the referenced children before the parents can be removed.

Chapter 11

Timekeeping

This section provides information about timesheets, including:

- Timesheet Basics
- Configuring Timesheets
- Integration with Timekeeping Systems

Timesheet Basics

Timesheet functionality allows your organization to more easily use the time information which is collected within the application as shifts are completed. This information can be exported to external timekeeping systems as needed. This functionality also allows mobile workers to review, accept, and sometimes change their reported time for work, travel, or non-work events (such as breaks, non-productive tasks and periods of unavailability) for their shifts. The system groups time on the shift into usage "time buckets" such as work, travel, idle, break, POU, depot, etc. For example, all travel can be grouped together, all inspections grouped together, all installations grouped together, and so on.

Timesheet functionality is optional. Global master configuration parameters must be specified to enable this functionality. When enabled, timesheets are created automatically at the end of the related shift for each period of time a mobile worker was part of the shift.

A Timesheet record is displayed in the Timesheet portal for review and acceptance, as needed. The base product resource timesheet business object assumes no further approval is necessary once accepted by the mobile worker and information is considered ready to be interfaced to an external system. You may introduce custom logic to require manager approval before timesheet information is interfaced.

The actual process to interface a timesheet record to an external system is not provided out of the box and has to be customized.

All generated timesheets create for a shift are listed in the Crew Shift Timesheet zone of the Crew Shift portal.. Refer to the user guide and online help for more information on timesheets.

Process Notes

Mobile workers will require training on how to review and accept timesheet information on their mobile devices and/or in the office as part of their shift completion tasks. If your organization allows workers to modify time information or you implement other custom interaction for users with their time, training will be needed for this processing as well. These details will depend entirely on your business practices and how you choose to implement this functionality.

Configuring Timesheets

Examine your business processes and work with your timekeeping department to make decisions regarding the mobile worker's interaction with the timesheets.

- What should be the time "bucket" associated with travel and work for the various task types a crew performs?

  By default travel and work time are summarized across all task types. If your business requires that travel and work are summarized separately for specific types of work you may introduce new timesheet time types and reference them on the corresponding task types.

- Can workers change reported time information if their actual work differs from what is reported?
- Should a manager manually approve a timesheet record a mobile worker has accepted before it is considered ready to be interfaced to an external system?

The crew process timesheet business object managed the process of generating timesheets for resources at end of shift.

The resource timesheet business object manages the information and business rules associated with a timesheet record for an individual resource.

Review these business objects and extend them as needed.

Complete the following configurations to set up timekeeping functionality within your organization:

- If a manager should manually approve a mobile worker's accepted timesheet record, create a custom algorithm based on the algorithm type "Generic Business Object Status Monitor (F1-GEN-BOMON)" providing the pending approval status code. Plug this custom algorithm at the reviewed state before the base algorithm. This causes the reviewed record to be transitioned to pending approval instead of processed state. A user would approve or reject the record as needed.
- Use the **Timesheet Type** portal to create records for a mobile worker timesheet type and a crew process timesheet type.
- As needed, create additional time buckets using the using the **M1-TimesheetTimeType** extendible lookup.
  Your implementation can change time bucket groupings as needed by adding more time buckets to the **M1-TimesheetTimeType** extendible lookup and referencing them on the various task types.
- Review task types to associate which applicable timesheet time types.
- Enable timekeeping functionality on **Master Configuration** under **Global Configuration** in the **Resource Management** section. Reference the timesheet type records you created. If values are not entered here, timekeeping functionality is disabled. This is also where you designate whether or not mobile workers are allowed to adjust the timesheet times as calculated by the system at the end of their shift. If this is allowed, the system presents the worker with the option to adjust values such as their travel time, work time, break times ect. when they verify and accept their timesheet.
- Add these timesheet types also to your deployment.. This is needed for the mobile application to find the necessary information when handling timesheets at end of shift.
- Ensure that the **M1-TIMSHT** batch control is running.
  This is mainly used to generate timesheets for non-MDT crews, but it also handles MDT crew timesheets that are waiting for task completion or remote messages to be resolved. If this batch job is not running, the functionality may not work properly. This batch control does not need to run frequently as it is only used to handle exception conditions.

Integration with Timekeeping Systems

You can add custom logic to export timesheet information to an external timekeeping system or other application.

A timesheet record in the processed state is ready for extract to an external system. Depending on how your system integrate with the external timekeeping system you can create a custom algorithm that on entry to the processed state sends a message to the external system. Alternatively the external system may poll for processed timesheet records.

Chapter 12

Handling Keys

The system can be configured to alert mobile workers that they are required to pick up keys (or access cards) before they can work on an activity. Obviously if key or card is needed to gain access to a work site or facility, the mobile worker cannot start an assigned activity without it.

This feature leverages depot functionality, so a distribution depot must be set up for every location where keys would be picked up. If your organization does not use depots in general, but will want to track key pickup in the way described, you must set up key related depots. After receiving the key information from the host system, such as the key ID and the depot where the key is located, the scheduler creates a depot task to pick up the key as part of scheduling the activity. If the same crew/mobile worker has multiple activities that require them to pick up keys at the same location, the scheduler creates one task to pick up all of those keys.

Activities that require keys are restricted to shifts that have the same logoff location as the location where keys are picked up. This is assumed to occur as part of the shift practices. The system does not validate whether or not a key was actually dropped off.

Activities already associated with depot to pickup or drop goods cannot use keys since an activity can only be in a single depot run. This means that there cannot be depot activities that require "goods" to be picked up from depot A and "keys" picked up from depot B. Then the activity would have to be in two depot runs to pickup both goods and keys.

Configuration for key handling functionality involves the following:

- Define a depot task type in the **Task Type** portal describing it as a task to pick up keys.
- Set up distribution depots for every location where keys would be picked up. Associate each with the depot task type for key pickup.

Please refer to the section on Depots for more information.

Chapter 13

Transfer of Goods

This section describes how to configure transfer of goods functionality, including:

Transfer of Goods Basics

Transfer of goods involves the process of transporting goods from one location to another. Not all businesses use this business practice, however, if they do, the system provides a way to manage goods, restrictions and capacities related to the transfer of goods.

Transfer of goods functionality involves the following entities:

- **Activities** - Describe transported goods and any restrictions on what cannot be shipped with the goods on the activity type.
- **Vehicles** - Describe any capacity limits for goods on the vehicle type.
- **Shifts** - Provides the opportunity for dispatchers to override any restrictions or limits on a specific shift.

If goods are transferred to and from depots, an activity can specify a list of valid depots where the goods can be picked up or dropped off. Crews can also be associated with a specific list of depots that they are allowed to visit. A dispatcher can also override this list for a specific shift. Refer to the depot section for more information.

Depots

If transferred goods are loaded or unloaded at a particular centralized location, your organization can set up depots to capture the information related to these locations. Depots specify details such as whether the depot is a pick up or drop off location, what types of goods the depot can handle, the physical address and the days and times that the depot is open.

Two types of depot operations are supported:

- **Distribution** - a depot where crews pick up goods for delivery. Distribution runs start at a depot where items for delivery are loaded and then subsequently dropped off at a number of drop off activity locations.
- **Collection** - a depot where crews drop off goods picked up from customers. Collections runs finish at a depot after the pickup of items at a number of activity locations.

  Note: If a depot location can participate in both collection and distribution runs then two depots need to be set up for the same location.

Depot Profiles Control Everything

Depot profiles can be set up to establish common settings for site delay, access time windows and service time windows across multiple depots. All other details are defined on each individual depot. If a depot refers to a profile, the settings established by the profile cannot be overwritten.

Depot Tasks

A depot task is a tool for tracking the crew's travel, arrival and completion of the depot visit. The system generates depot tasks as needed when planning depot runs to and from depot centers to accommodate depot activities. A depot task is dispatched to a crew, just like an activity, and its status is updated and tracked in the same way.

The scheduler creates a depot task using the task type defined on the depot or its depot profile and is specified it used the task type on the master configuration.

You may want to set up designated depot task type for different depots to better convey via the task type description the types of goods involved.

Depot Breaks

The system can also be configured to designate breaks which can only be taken at a depot. If a shift is not scheduled for depot work, the break is scheduled as a standard break.

Configuring Transfer of Goods Entities

The following guidelines provide a basis for transfer of goods configuration

Business Object	Description
M1-DepotBreakType	Depot Break Type Handles breaks if they are to be taken at a depot. If you are using only this business object to define break types you may want to update the M1 Break Type business object to not allow new instances.
M1-DepotRelatedActivity	Depot Related Activity Activities that perform distribution or collection of goods. If you are using only this business object to define activityies you may want to update the M1 Activity business object to not allow new instances.
M1-DepotRelatedActivityType	Depot Related Activity Type Handles types of activities to be performed at a depot such as delivery and collection. Reference the Depot Related Activity business object on these types of activities. If you are using only this business object to define activity types you may want to update the M1 Activity Type business object to not allow new instances.

Business Object	Description
M1-DepotRelatedShift	Depot Related Shift Handles shifts that are worked at a depot. This business object is a subclass of the base crew shift business object with additional support for vehicle capacity limits and depot restrictions. Reference this business object as the shift business object when defining your shift types. If you use only this shift business object for all your shifts you may want to mark the M1 Crew Shift business object to not allow new instances. Note that for template purposes the standard shift template business object should be used on the shift type even for depot related shifts.
M1-DepotSinglePersonCrew	Depot Related Single Person Crew Handles single person crews that operate from a depot. Reference this business object when defining crew types. If all crews use this business object you may want to mark the other crew business objects to not allow new instances.
M1-VehicleTypeWithCapacity	Vehicle Type with Capacities This business object is a subclass of the M1-VehicleType business object with additional support for vehicle capacity limits. If all vehicles use this business object you may want to mark the other vehicle business object to not allow new instances.

Define Vehicle and Depot Capacities

Capacities should follow these parameters:

- If you are using cumulative capacities to describe goods, such as weight, volume etc, override the descriptions of up to 5 capacity types defined in the lookup M1_CUMULATIVE_CAPACITY_FLG to enable them.
- If you are using non cumulative capacities to describe goods, such as size, override the descriptions of up to 5 capacity types defined in the lookup M1_NONCUM_CAPACITY_FLG to enable them.
- If you are using maximum cumulative capacities to describe maximum load, such as maximum weight, maximum volume etc, override the descriptions of up to 5 maximum capacity types defined in the lookup M1_MAX_CAPACITY_FLG to enable them.
- If you are using maximum non cumulative capacities to describe maximum load, such as maximum size, override the descriptions of up to 5 maximum capacity types defined in the lookup M1_MAX_NONCUM_CAPACITY_FLG to enable them.

Scheduler Configuration for Depots

- Set up corresponding scheduler configuration costs for applicable cumulative and non-cumulative capacity types. For example, if you defined "cumulative capacity 1" as "weight" then make sure you provide the cost for weight in the "cumulative capacity cost 1" parameter.
- Set up depot related parameters as needed.

Chapter 14

Contractor Management

This section provides information about contractor management functionality, including:

Contractor Management Basics

Contractor management functionality supports having outside contractors or a supplemental workforce completing work for your organization. There are two models for contractor resources:

- **Shift model** - In the shift model, contractor resources are set up, configured, and scheduled as if they are internal employees, vehicles and crews.
- **Capacity model** - In the capacity model, contractor companies are assigned work based on agreed upon capacities of work of a certain type, in specific areas and for a period of time. Work capacity maybe specified as a number of hours or a number of activities. Contractor resources are fully managed and scheduled outside of Oracle Real-Time Scheduler.

This optional functionality can use both models in one implementation since each contractor is set up and managed on an individual basis.

Capacity Model

Company users set up work capacities to reflect agreed upon agreements between your company and the contractor company.

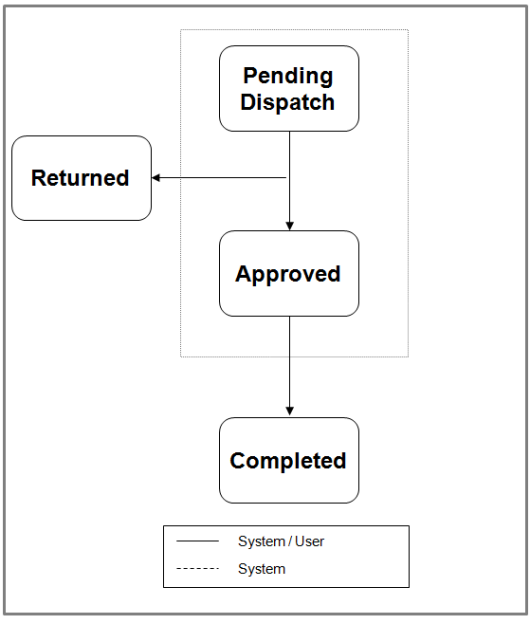
At the predefined notification lead time capacities are closed and all work scheduled to them are committed to the contractor. An assignment is created to reflect this commitment.

Once committed the system waist for the contractor to approve or reject taking on the activity committed to them.

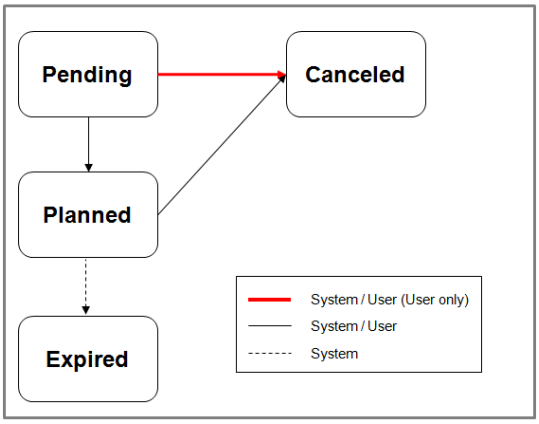
If the contractor company is fully integrated with your company the system sends the committed activity to the contractor via a web service call and they should respond by either rejecting or accepting the activity. If not integrated the contractor has to have access to the Contractor Work Management portal to approve or reject activity commitment.

An assignment representing an external capacity contractor's commitment for an activity follows a different lifecycle than a crew assignment but it serves the same purpose with respect to capturing completion information.

The following diagram depicts the lifecycle of a capacity contractor assignment.



Capacity contractor assignment states are depicted in the following diagram:



Configuring Shift Based Contractors

Create a contractor Resource Group Type for the shift based contracting model. Use the Contractor Type business object and set group class to shift based.

For each shift based contractor company, complete set up for their resources as follows

- Set up a resource group of the shift based resource group type.
- Set up each contractor mobile worker as user s in the system. It is very important that you reference the resource group defined for the contractor company they work for in the **User** portal **Miscellaneous** tab.
  It is recommended to set such users with the My Calendar home page.

  Only mobile workers should be provided with user access to the system. All other roles are assumed to be company employees.

  Your organization should define contractors in user groups then provide them with access only to the certified transactions that apply data security. Please refer to Contractor Security for more information.

- Set up mobile workers, vehicles and crews for each contractor resource and reference the contractor company they belong to.
- Set up MDTs as needed if these crews would be using the mobile application in disconnected mode.
- Create crew specific shift templates or subscribe these crews to common templates to generate their shifts just as you would for company crews.

Configuring Capacity Based Contractors

Create a contractor Resource Group Type for the capacity based contracting model. Use the Contractor Type business object and set group class to capacity based.

Capacities can be defined by number of hours or number of activities. Create a Capacity Type record for each option application to how you work with your capacity contractors.

For each capacity based contractor company complete the following configuration

- Set up a resource group of the capacity based resource group type.
  Note: review the contractual agreements with the contractor company to properly reflect expectations around notification lead time and approval time. Make sure the contractor record is associated with service areas they are eligible to work in.
- Use the **Capacity Weekly Template** portal to establish shift-like templates that reflect the agreed upon work capacities between the contractor company and your company.
- If the contractor company integrates with your application via web services, customize the message exchange between their services and the base product services. Refer to the Web Services chapter for more information. No user access should be provided to the contractor employees.
- If the contractor company does not integrate with your application, provide user access to some of their employees to so they can manage work committed to the contractor company via the **Contractor Work Management** portal.

Your organization should define contractors in user groups then provide them with access only to the certified transactions that apply data security. Please refer to Contractor Application Security for more information.

Contracting Work

Settings on activity type indicate whether the activity should be worked by company resource only, any internal resource, whether company or can be assigned to any resource.

This can be overridden on the activity.

The scheduler would not violate this restriction unless work is manually allocated or dispatched.

Review **Activity Types** to determine which types of activities can be worked by contractors.

Scheduling Preference

By default scheduler prefers internal resources, company or shift based, over external contractor capacities. If you need to set scheduling preferences between contractor companies and your company via Eligibility Rules, also define your company as a resource group as follows

- Create a company Resource Group Type by using the Company Type business object. You can only create one company resource group type.

- Create also a company Resource Group based on this type. You can only create one company resource group.

  Use the **Contractor Eligibility** portal to set up scheduling preferences between the various resource groups, contractor companies and your company for a particular service area and service class.

These preferences are stamped onto each activity that is eligible to be worked by contractors based on its service area and type of work and sent to the scheduler along with activity details. Information is stamped when the activity is created or rescheduled. If the system detects a change to eligibility rules while the activity is still being scheduled it updates the preferences to reflect the change next time the activity is monitored by the task monitor batch process.

Logic to determine contractor scheduling preferences resides in a plug-in "Contractor Eligibility" associated with the activity business object. You may implement a custom algorithm to accommodate your business requirements as needed. Refer to B: Algorithm Entities for more information about this algorithm.

Preferences are used by the scheduler when scheduling the activity. Scheduling cost is associated with not honoring this preference option. For example, if your preference is capacity then it would be less expensive to use a capacity over an internal resource.

Review your scheduling preferences and set eligibility rules and cost control parameters accordingly.

Contractor Security

Secure Specific Functions

All BPA scripts are released with a default application service code. Implementations must decide which script to secure and associate with custom application services.

The following BPA scripts represent functions related to shifts, capacities and activities and are available on any portals that a contractor user may have access to.

- M1-AlocAlShf (Allocate All Assigned Activities Of A Shift / Capacity)
- M1-OpenShift (Open A Shift / Capacity)
- M1-CloseShft (Close A Shift / Capacity)
- M1-CrShfClAl (Crew Shift / Capacity - Open/Close/Allocate)
- M1-CrAsstAct (Create Assist Activity)

If you have already secured these BPA scripts with a custom application service, make sure you do not provide contractor users with access to your custom application service.

If you have not secured them you should do so using the base application service. CDF Execution Service(M1-DFLTCDF).

Defining Contractor User Groups

Shift based contractor users and capacity based contractor users need to have access to different parts of the application.

It is important that contractor users only access their data. The base product secures specific transactions only to restrict the data accessed by a contractor user such that they can only view their data.

Warning! Do not provide access to any transaction that is not in the secured list.

At a minimum there should be one user group that includes shift based contractor users and another for capacity based contractor users.

Shift Based User Group

In the shift based model only mobile workers need to have access to the system. All other user roles are assumed to be company users.

The shift based contractors user group should have access to the following:

Portals

- Activity search portal and all its query options
- Activity maintenance portal
- Assignment maintenance portal
- Crew Shift search and all its query options
- Crew shift maintenance portal
- Break / Non-Productive Task maintenance portal
- POU maintenance portal
- Depot Task maintenance portal
- My Calendar portal

Business Objects

Activity. Only provide read access. Do not provide access to actions add, change, cancel, delete, allocate, unassign, defer, assist etc.

POUs (Only provide read access)

- Assignment, Break, POU Task, Non-Productive Task, Depot task. Do not provide access to actions add, delete actions.
- Crew Shift. Do not provide access to actions add, cancel, delete, force logoff, open, close, allocate all.

Capacity Based User Group

If your company integrates with the contractor company's application via web services there is no need to grant access to any contractor user.

When applicable, part of your business practice might be to provide contractors access to your Oracle Real-Time Scheduler system via the **Contractor Work Management** portal so that they can access the work that has been to assigned to them through your organization and to approve and decline work.

You must configure security settings so that when a contractor accesses the system, they can only see their work and not the work of other contractors or for your company.

The capacity based contractor's user group should have access to the following:

Portals

- Contractor Work Management portal
- Activity search portal and all its query options
- Activity maintenance portal
- Assignment maintenance portal
- Capacity search and all its query options
- Capacity maintenance portal

Business Objects

- Activity. Only provide read access. Do not provide access to actions add, change, cancel, delete, allocate, unassign, defer, assist etc.
- Assignment. Do not provide access to actions add, delete actions.

- Capacity. Do not provide access to actions add, cancel, delete, open, close, allocate all.

Defining Contractor Users

Irrespective of the contacting model used, contractor users should be defined in the system referencing the contractor company they belong to. This is defined on the Miscellaneous tab on the User portal.

> **Warning!** If not associated with a contractor resource group they are considered company users and allowed access to entire data.

Associate each user to the corresponding contractor user group.

Integration with Contractor Systems

Define an **External System** record to describe the integration with the contractor's system. Refer to XAI configuration for more information.

Next, reference this External System on the contractor's resource group.

Integration with a capacity contractor system involves the following message exchanges:

- When an activity is committed to a capacity contractor the system sends the activity details to the contractor.

  The system creates a Synchronization Request in the pending state which is then processed by the Synch Request monitor batch process. Ensure the batch process is run often enough to process these messages in a timely manner. When processed the synchronization request sends an outbound message to the contractor's system.

  The synchronization request business object that manages the sending of the message has to be defined on the "M1-ExternalSystemMessages" extendable lookup for the specific external system code. Define an outbound message type for such a message.

  Populate this outbound message type as a schema contestant of type "m1ActivityToContrMsgType". This is defined on a Feature Configuration of schema constants type.

- Any change made to activity details that were sent to the contractor causes another message to be sent with the updated information.

- The contractor then has to respond by accepting or rejecting the commitment. They need to call the "**Process Activity Commitment By Contractor (M1-ActivityCommitByCont)**" XAI inbound service.

  If not approved in a timely manner, i.e. within the approval time defined for the contractor, the system automatically assumes they have rejected the commitment and excludes this contractor from the activity allowing it to be rescheduled as needed.

- When work is complete by the contractor they should send in completion information for the activity using the "**Process Activity Completion By Contractor (M1-ActivityCompleteByCont)**" XAI inbound service.

Chapter 15

Mobile Communications Platform (MCP) Configuration

This section provides information relating to the Mobile Communications Platform (MCP), including:

Configuration Tools

Standard framework configuration tools, such as business objects, scripts, UI maps, algorithms, etc., are used to create, extend and customize the mobile application. etc. Refer to the Oracle Utilities Application Framework Administration Guide for more information.

It is important to note that the mobile framework is a slightly scaled-down version of the Oracle Utilities Application Framework.

Business Objects

Entity maintenance is only performed via interaction with business objects. The business object layer does not interact with an underlying maintenance object layer, but rather is based on a completely different data model architecture. This means that all business rules to be performed on the mobile application need to be defined as business object rules.

The business object framework is limited as follows:

- Automatic state transition is not supported.
- Transitory states are not supported.
- Only two plug-in spots are supported.
  - MCP Enter
  - MCP Post Processing

  Refer to Appendix B: Algorithm Entities for more information about these plug-ins.
- BO invocation for read, add, replace, and delete actions is supported.

The same business object definition that controls an entity on the server application also controls it on the mobile application; however, different business rules apply. On mobile, entity-specific business logic must be triggered by these MCP-specific business object plug-in spots only.

Scripting is the only supported programming language for these plug-in spots and java scripting for user interface rules. Service scripts and plug-in scripts are limited to XPATH version 1.0.

As mentioned earlier, the data model architecture is completely different from the server framework. Instead of individual maintenance objects, the mobile framework is based on a single XML data storage for business object instances. This also means direct access to tables for query purposes is not possible. All queries need to be performed by calling the MCP Query mobile framework business service. Refer to that business service for more information.

Custom Business Services

Custom business services can be created to allow you to create more complex or performance sensitive logic. For example, these business services can be enabled to allow for field billing, scanning, printing, other digital captures and so on.

Details on environment setup, implementation requirements, and deployment can be found in Appendix E: Mobile Communication Platform Custom Business Services Development Setup Guide.

User Interface

There is no notion of portals, zones or BPA scripts on the mobile device. Service scripting syntax in the mobile framework supports a "Terminate with Map" command that allows user interface interaction. This is the only means of presenting data to and obtaining data from a user.

Different devices may call for a different UI map to maintain and display an entity's details. For example, a handheld device has less real estate than a laptop. To accommodate differences in screen size, the MDT type record provides a display option attribute, which specifies the type of display (such as Windows Mobile VGA or Windows XVGA). Further, each business object that is maintained by the mobile application, like task and shift business objects, has an option where you can specify the mobile script to use for each supported device display option.

Navigating Back to a Previous Map

When navigating to a UI Map with this tag, the tag tells the MCP to push the previous UI Map name and data onto the stack of backable maps. This is in case users execute a "back" at some point. In this case the MCP displays the UI Map name and data from the stack and navigates to it. If this tag is not on the UI Map being navigated to, then the backable map stack is cleared.

For example:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/
TR/html4/strict.dtd">

<html >

   <body oraMCPSupportsBack="true">

    …

    …

    </body>

</html>
```

Clickable Phone Numbers

You can use the **oraPhone** and **oraPhoneContact** attributes to identify phone numbers on UI Maps.

For example:

```
<td oraPhone="true" oraPhoneContact="boGroup/customerInformation/
contactName" oraField="boGroup/customerInformation/mainPhone"></td>
```

Device vs. Server Time Zones

The mobile framework supports the user working in the local time zone for their device while at the same time managing the conversion to and from the base time zone when communicating data to and from the server.

Refer to the Time Zones section for more information.

Navigation and Display

This section describes configurable aspects of the mobile application work flow and display.

Everything Starts with the Initial Script

When a user logs on to the mobile application, the mobile framework invokes the Initial Script defined for the specific deployment on the device. Refer to the deployment section in the user guide for more information on deployments.

The base package Initial Script demonstrates how a crew obtains their shift for the day, starts the shift, and then is presented with their tasks for the day. The crew can work each task and use other actions to manage their shift.

As mentioned earlier, the maintenance user interface associated with each task and shift business object can easily be replaced with your custom user interface, as these are defined as business object options. If you need to customize common parts of the mobile application that are not business object specific, clone the Initial Script and adjust it as needed.

Indicator Bar

The user interface on mobile devices includes an indicator bar which shows various indicators such as new email and connectivity. You can introduce your own indications by doing the following::

- Add the special icons to the "Mobile Device Indicator Bar" extendable lookup.
- Add custom logic to the mobile application to properly add or remove the indication from the bar as needed.

  To this you will need to call the MCP Indicator Bar Manager business service.

The Tool Bar

The mobile application displays a toolbar below the application indicator bar. The toolbar is available from every application screen, but the buttons that are displayed at any time vary depending on the current application screen you are viewing.

General Actions Menu

The **Task List** user interface supports an **Actions** icon that when clicked presents the user with a list of actions a crew can take during their shift, for example changing their primary function, changing crew allocation, going offline, completing their shift etc.

To add more custom actions

1. Copy the base actions script and UI map to create your custom script and map. To ensure upgradability, include the base script as your custom script's schema.
2. Enhance them to support your custom actions.
3. Copy the base initial script and set it to call your custom actions script.
4. Use your custom initial script in your deployments.

Task Actions Menu

When on a task related user interface a menu actions icon appears if configured on the task business object. The base activity business objects are configured with a BO option to manage activity related actions such as creating a new activity for the same location, capturing contents for this activity etc.

To add more task related custom actions

1. Copy the base actions script and UI map to create your custom script and map. To ensure upgradability, include the base script as your custom script's schema.
2. Specify your actions script as a business object option with a higher sequence number than the base actions script.

Geographic Map

If the device is configured to support mapping, a geographic map icon appears on the task list user interface as well as on task related maps. You can configure special theme layers to be displayed on the maps which users can access for tasks on their devices. For example, you might configure layers to show the location of company assets. Different types of crews can be associated with different applicable layers.

- Refer to customize device capabilities to enable mapping for a specific MDT Type.
- Refer to the Map Layers plug-in spot to configure custom layers to appear on the map.

Remote Script Invocation

Data and updates sent to the crew, as well as updates and completion data received from the crew, rely on the Remote Script Invocation (RSI) mechanism supported by the mobile framework.

Data exchange is achieved by calling a framework business service to place an asynchronous request to invoke a service script and its corresponding data on a remote application. The system provides both server and mobile device versions of this service. The server version places the request to be performed on a specific MDT, whereas the mobile device version places the request on the server. A synchronous RSI call to the server is also supported.

Messages Sent to the Crew

Various algorithms implement logic to send messages to the crew's mobile device. For example, the Task Synchronization algorithm is responsible to dispatch tasks and send updates to dispatched tasks.

A call back notification functionality is supported for RSI messages issued from the server only. With this functionality, the process posting the RSI message to the crew may request to be notified if delivery of the message to the crew takes too long. For example, this may be used to raise an alert when an activity takes too long to dispatch to a crew.

Messages Received from the Crew

A synchronous form of an RSI call may be issued from the mobile device only, such as when the crew request and start their shift for the day.

On the server, messages sent from the crew are processed by the Remote Script Invocation installation plug-in. This allows a better handling when an application error occurs. The base algorithm persists the message as a crew message business object and if an error is encountered while processing the message a To Do Entry is raised for a user to correct the issue and reprocess the message. Refer to Appendix B: Algorithm Entities for more information about this plug-in.

Remote messages in an error state are periodically retried by the remote message monitor batch process. This process should be scheduled to run in a timely manner. Refer to Base Package Batch Controls to Run Periodically for more information.

Mobile Devices

Mobile data terminals (MDTs) such as laptops, cellular phones, tablets and other mobile devices may be used to interface with the system via the mobile application.

An MDT Type defines attributes common to MDTs of a certain type. For example, it specifies the Mobile Communication Platform compatible with devices of the type. Many features and settings depend on the supported platform.

The mobile application can be used in a disconnected or connected mode stated by the Mobile Communication Platform selected.

- Disconnected mode: In disconnected mode, the mobile application and data reside locally on the mobile device allowing the crew to work offline as needed. The actual device used has to be registered in the system as an MDT and has to be an officially supported type of device.
- Connected mode: In connected mode neither data nor the mobile application resides locally on the mobile device. They reside on the server and the user must be connected to the server at all times using their browser to access the mobile application. In this mode the actual physical device used to access the mobile application is irrelevant and therefore registering the specific device in the system is not needed. The system still utilizes the notion of an MDT but it generates virtual MDTs as needed using an MDT Type set with connected mobile communication platform.

  Note: This should not be confused with the server application. This is the mobile application presented in a regular internet browser.

In either connection mode, when a user starts a shift, the system links both the MDT and the user ID to that shift and uses this association to communicate activity and shift data to the appropriate device.

Refer to the installation guide, online help and user documentation for more information.

Device Capabilities

The capabilities section on the MDT Type allows you to define the various features or functions supported by mobile devices of this type, such as capturing photo or sound, attaching supporting documents or launching a geographic map.

The MDT Type also controls the types of attachments that can be opened on a device, and sets a limit for the attachment size. When a user attempts to download an attachment, the system validates that the total size of the download plus what is already on the MDT will not exceed the maximum for the MDT.

Note: If a capability is not explicitly enabled on the MDT Type, the corresponding function is not available on the device. Review your device types to ensure their platform is supported and enable functions to represent true device capabilities.

Mobile Device Registration

You must create an MDT record for each physical device that will communicate with the main server application in disconnected mode.

The MDT's type has to be of a compatible mobile communication platform.

The first time a mobile user runs the mobile application from a new device, they will be required to register the device. The system will ask for the MDT Tag and the URL of the server application. (The MDT Tag is a unique, user-defined identifier that is assigned to the device when the MDT record is created on the server application) As part of the registration process, the system places the device's unique ID in the MDT record on the server. The unique ID is derived from the IMEI for mobile devices or from a motherboard serial number for Windows laptops or tablets. Once the device is registered, the mobile user will be able to log on to the application from that device without further prompting.

During the logon process, a user provides their user name and password for authentication. If authenticated, the system then looks for the most recent version of all active, qualified deployments based on the user's user group, language, and MDT type. The mobile application cannot be deployed to a device unless it has been successfully registered.

Refer to Mobile Application Configuration for more information.

Connected Mode Configuration

To access the mobile application in connected mode the user needs to reference your organization's mobility server URL in a browser, as they would for any website, and enter their username and password.

In most cases the mobile URL is derived using your regular server URL with "mobility/Login.jsp" appended. Example: http://[your server]:[port]/mobility/Login.jsp.

After the user is authenticated, the system allocates an MDT record of a connected platform MDT Type from a pool of available records to the user. If all connected MDTs are already in use by other users, the system duplicates an existing connected MDT record to create a new MDT record and allocates it to the user. Once allocated to a user, the MDT is stamped with that user and is not available to another user. When the shift is completed and all messages are processed transactional data is cleared from the connected MDT data store on the server and the MDT is returned back to the pool and made available to all.

Logic to allocate an MDT to a user in corrected mode resides in the Determine Connected Device algorithm defined on the installation options. Refer to this algorithm for more information.

To configure connected mode:

- Create a generic MDT Type defined as using the connected MCP platform.
- Create at least one connected MDT record based on this type in advance.

  Note: The system would created more MDTs as needed as users access the mobile application in connected mode but it is recommended to create a pool of such MDT records for a more efficient MDT allocation process. At the end of the shift, the MDT is returned back to the pool so the pool should be large enough to accommodate concurrent shifts using this mode.

  Also note that the Display Option is automatically switched on Connected MCP based on the device resolution reported by the browser. So the selection on the MDT Type is ignored for Connected MDTs.
- Ensure that the "Stale MDT and RSI Cleanup (M1-CLMDT)" batch process is run periodically. Refer to the batch section for more information.

The following limitations apply:

- GPS data will not be captured or sent to the server. Connected MCP users are tracked in the same way as non-MDT crews.
- Attachments, capture picture and capture sound are not supported.

Mobile Device Log Files

In disconnected mode, logs from the runtime mobile application are written to a file stored locally in the mobile application's logs folder. When the content in the file exceeds a configurable limit, the currently used log file is renamed, appending the current date and time to the file name, and a new file is created. When the number of log files exceeds a configurable limit, they are moved to the archived log files folder on the device. Archived log files are kept for a configurable number of days before being purged. Logging parameters (log file size, file count, archival days, etc.), are defined on the Mobile Data Terminal (MDT) record. When a device logs on to the mobile application, these settings are read from the server and saved to a local properties file.

The system does not send logs from the MDT automatically. These log files are only sent on demand, usually for debugging and tracing purposes. To send log files to the server, navigate to the device's **MDT** portal in the server application and then click the **Get MDT Log Files** button. This initiates a request to the device to send the active log files to the server. On the server, log files are stored in separate folders for each MDT and may be viewed using the **Log Files** tab on the **MDT** maintenance portal.

> **Note**: The MDT log files described in this section are not to be confused with the business audit log records associated with entities such as tasks and shifts. The latter are not files, but rather part of the business entity's completion data that are automatically sent back to the server when work on the entity is completed.

The logging level is specified in the MDT record.

In connected mode, log records are captured on the server log uniquely identified by MDT and user. Being already on the server they are readily available on the **Log Files** tab.

Mobile Data Cleanup

Review the **M1-CLMDT** batch control and set the parameters to ensure that it is set to clean up the RSI and state of the connected MDT transactional data as required by your business practices. By default the logs are displayed for the current MDT and the logged-in user. If the MDT user is different from the logged-in user then the User ID has to be entered on the **Log Files** tab.

Chapter 16

Mobile Application Configuration

Mobile application development is completed by creating application components, such as business objects, UI maps, and service scripts, using the same development tools used for the main application. Once the mobile application components have been developed and tested, they must be packaged together along with all necessary control table records to be delivered to a mobile device terminal (MDT).

A deployment refers to a specific cut or build of a mobile application of a specific deployment type configuration and for a specific language. It is a packaged set of application components needed to carry out mobile application functions using a MDT.

Refer to online help and the system administration guide for more information on deployment configuration. You may also refer to the demonstration database for examples of deployment definitions.

This section provides information about deployments including the following:

Defining Deployment Types

This section summarizes steps involved in the deployment process. Refer to the online help or user documentation for instructions on how to define each entity. Use the embedded help to display field descriptions.

- Set up a deployment type for each type of application you need to support. You must take into account the screen size such as whether the application is designed for a hand-held or laptop.
- Create deployment parts to represent subsets of your custom deployment items based on their type or usage. The base product provides pre-configured deployment parts that include the base mobile application components. You only need to configure parts to include your custom entities.

  You should share deployment parts across deployment types when applicable to prevent redundancy and improve management of deployment configuration.
  For example, use a designated deployment part for control data, another for items that

are common across platforms, another with user interface items for a specific platform etc.

To simplify the collection of items in a deployment part it is recommended that you create an export bundle per deployment part so that implementers can add their new or changed items directly to the corresponding bundle using the dashboard zone. Once items are collected the bundle can be uploaded to the deployment part. You can always maintain items manually on a deployment part..

- Associate deployment parts to appropriate deployment types.
- It is recommended that you create designated custom implementation message categories for messages your application uses on the mobile device.

  If you are using the questionnaire style Procedure Type, it is recommended that you create designated custom implementation message categories for different types of procedures. Refer to the Procedure Types and Procedures Basics section for more information.

- You should have already identified the device platforms your mobile applications should support and define corresponding MDT Types. Associate MDT Types to appropriate deployment types.
- You should have already identified the user groups authorized to use each type of mobile application. Associate user groups to appropriate deployment types.

When you have finished defining deployment entities, you must run a batch process to create the deployment.

Creating a Deployment

A deployment can only be created using the batch deployment process, **M1- DPLOY**.

Once your deployment type setup is complete, submit the batch to create a new deployment for each specific cut or build of a deployment type for a specific language.

The status of the new deployment is **Created**. You must manually activate the deployment before it can be deployed to an MDT. Use the **Deployment** maintenance portal to view existing deployments and change the status (activate or inactivate).

When a deployment is created for a deployment type and language, all the open "Deployment Out of Date - TODOs" are detected and automatically closed for the same configuration.

Version Control

The MCP communication architecture includes both client side and server side elements. These must be compatible or else errors will occur.

To enforce compatibility, the system validates the MCP version as follows:

- When the MDT attempts to connect to the server, the system versioning check validates whether or not the MCP Version on the MDT is compatible with the MCP version on the server. If the versions are not compatible, the device is not allowed to work online and no communication is sent to/from the server. In this situation, the MDT must be upgraded to the correct version before it can work online again. If there was data on the MDT, the user can continue offline and manually collect any data they need before the MDT is upgraded. If a user tries to log on with an incompatible MCP on their device, they receive a message.
- When a deployment is created on the server, it is marked with the version number of the server side MCP Components that created it. Then when a MDT logs on to download a

deployment, only deployments with compatible MCP Version can be downloaded. Both the server and the MDT are flagged if the deployment is out of date.

Warning: Before the server is upgraded, all MDT's should be logged off and all shift information on them should be successful completed for end of shift processing. This will prevent the need for manual data copying. You should never do server side updates until all MDT Crew Shifts have been completed. Use the crew shift search query option to find all started shifts and coordinate their successful completion before server side upgrade.

Out of Date Deployments

Deployments become out of date when a new MCP version is released, the deployment type configuration changes, or any underlying application component that is part of the configuration changes.

As a general part of accepting a new product release, new deployments are required. When the release is applied, all existing deployments are marked as out of date so that it is clear which deployments need to be recreated.

The **Deployment Evaluation and Purge (M1-DPUTD)** batch process evaluates existing active deployment records and marks those that are out of date if the program detects any metadata changes. The background process also purges out of date deployments that are older than a specified number of days. Refer to the Chapter C: Batch Controls for more information.

When the system marks an active deployment as out of date it also creates a To Do entry in order for a user to create a new deployment as needed.

Mobile users are warned at login to their mobile device when they are use an out of date deployment. The user is allowed to continue but is advised to contact their administrator to synchronize their deployment.

Ensure the batch process is run on demand or periodically based on your business needs to avoid situations where out of dates deployments are being used.

Downloading Deployments

When the mobile user logs on to the mobile application, the system looks for the most recent version of all active, qualified deployments based on the user's user group, language, and MDT type.

- If no deployments are found, the system displays an authorization error message and prevents the user from continuing.
- If only one deployment is found and it is the same as the one currently deployed, then the deployment is not downloaded.
- If only one deployment is found and it is newer than the one currently deployed or there is none currently deployed, then the deployment is downloaded to the MDT.
- If more than one deployment is found, the user can select the one they want to download. Once the application components defined in the deployment are downloaded to the MDT, the user can begin using the system.

  Note: An MDT's current deployment ID is stored in the MDT.properties file in the Data subfolder of the mobile application's program directory.

  Note: If unprocessed data exists on the device when a user downloads a new deployment, the system warns that the unprocessed data will be deleted if the user continues with the deployment.

Mobile Application Testing

The system provides tools to help implementations test their mobile application.

This provides improved efficiency in building mobile UI maps and other mobile application components to help reduce implementation time lines.

The **M1-MCPDevTools** deployment part has to be included in the deployment type being tested for the tools to be enabled.

During implementation, the development tools deployment part can be included in any deployment type to enable testing. When ready for production, the deployment part should be removed from the deployment types to disable the developer features.

> **Warning!** When ready for production, the deployment part should be removed from the deployment types to disable the developer features.

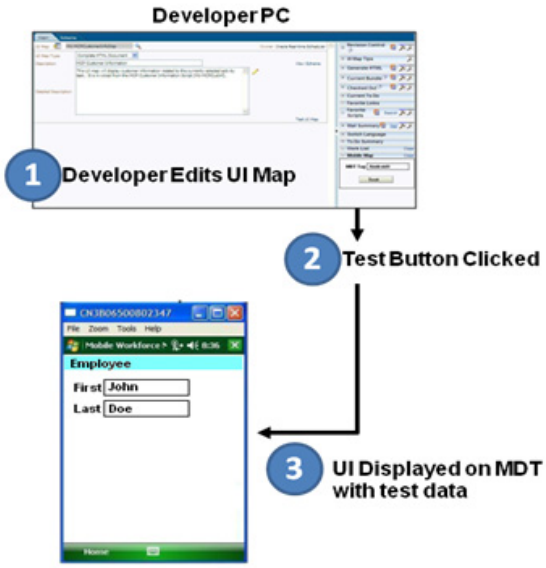
UI Map Testing

This tool allows a user to quickly test their mobile UI map on the device without going through the entire mobile application data setup. Launch a mobile application on any MDT to use the tool.

The deployment may be based on your actual deployment type or it can be a standalone deployment type that only includes the **M1-MCPDevTools** deployment part.

If using a testing deployment type, use the **M1-MCPIUITST** service script as the Initial Service Script.

The following image depicts how the tool works:



Steps to Test a UI map

1. Launch your mobile application.
2. Access the UI Map portal and make the necessary edits.
3. You may also provide test data as part of the UI map XML element, just like you do when testing server UI maps.
4. Enter your MDT Tag in the Mobile Map zone on the dashboard, and click the Test button. If you typically use the same MDT ID for testing you can reference it as a characteristic on your user record and the zone would default it when displayed .

A new window opens to display the changes on the MDT using test data if defined on the UI map.

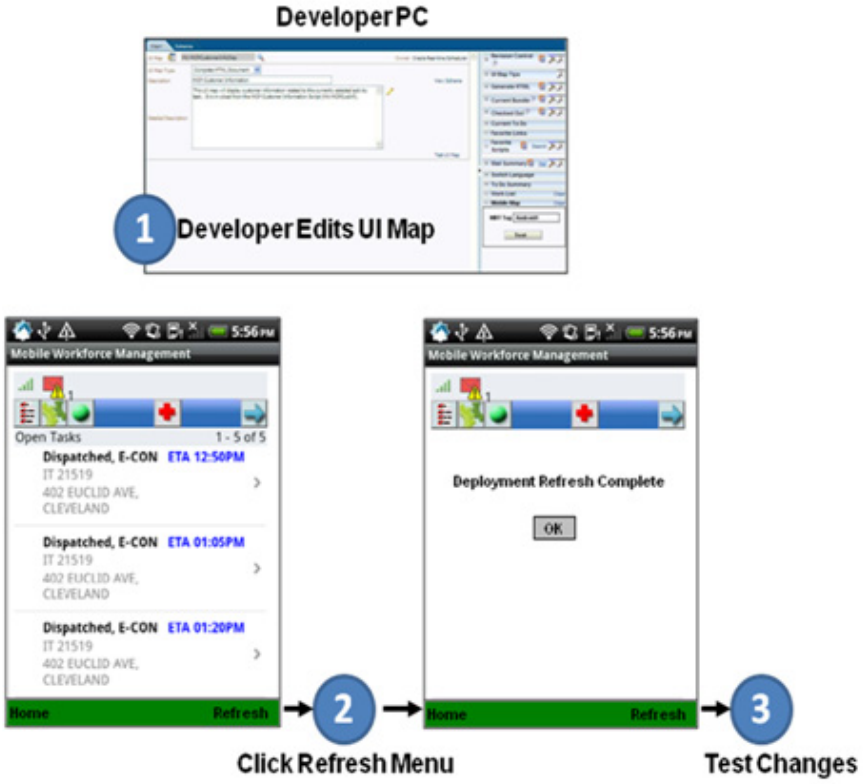
When this is initially launched, the MDT needs to have already downloaded a deployment and be displaying an application screen (not one of the initial MCP screens such as Register, Logon, or Deployment). This will allow you to logon to the MDT and the device will be ready to receive RSI messages from the server to test UI Maps.

This feature is not yet supported for the Connected MCP platform.

Refresh the Mobile Application

This tool allows a user to refresh all items in the deployment currently used by their MDT.

The following illustration depicts the main function of the tool:



Steps to Test a UI map

- Launch your mobile application.
- Change a mobile application item, for example a UI map.
- Click the **Refresh** button on the mobile application. This process rebuilds the deployment and sends it to the mobile device. It does not create a new deployment but rather update an existing deployment capturing the time it was updated.

  This one click approach can be used instead of re-creating the deployment on the server each time there are metadata changes done. This will re-generate the deployment on the server and replace the deployment already present on the MDT.

Alert Types

Specific alert types must be configured to support MCP functionality.

Panic Alert

Crews can select the panic button on their mobile device to alert the dispatcher that they are in distress. If the mobile device has the capability, the alert is sent with the GPS location of the crew at the time that the panic alert was issued to better locate the crew on the map.

This functionality requires:

- The panic alert type configured according to the following:
  - No creation algorithm
  - Allow manual close
  - Allow snooze and recycle
  - Use the "snooze forever" evaluation algorithm
  - Set to highest priority
- Reference the panic alert type on the Global Configuration and in the mobile application section specify the number of seconds for the countdown.

Timed Event Alert

From the general actions menu, crews can create events with an expiration date to indicate that the dispatcher should check on them if the event expires. This may be used if the crew feels they may be in a dangerous or hazardous situation and would like higher visibility from the dispatcher.

Configure the alert type for a timed event according to the following:

- No creation algorithm
- Allow manual close
- Allow snooze and recycle
- Use the "snooze forever" evaluation algorithm
- Set to highest priority

Chapter 17

SMS Messaging

Crews may use SMS messaging to carry out their scheduled work. SMS crews communicate via text messages sent from their cell phone.

Logic to process incoming SMS messages and send messages back to the crew resides in the following installation options plug-ins:

- Remote Script Invocation
- SMS Receive
- SMS Send

Refer to Appendix B: Algorithm Entities for more information on these plug-ins.

The SMS plug-ins use a phone number to send and receive the messages. Configuring specific phones to interact with the application depends upon the service provider, but generally relies upon SMS settings on the phone itself. Out of the box, the plug-ins are coded to transform the messages to BPEL. BPEL then calls the provider to send and receive messages. Also note the following:

- XAI Inbound Service, F1-SmsReceive receives the SMS message.
- To send an SMS using the base algorithm that uses BPEL, the "SMS Send Configuration" feature configuration has to be configured with the BPEL server and service details.

Sample Scenario

The following scenario illustrates how SMS messaging might be used:

- A crew sends a text message of "Logon". The system logs the crew shift on and sends details of the first task.
- As the crew works the task, they send messages to communication their progress, such as "1" to indicate that they are en route to the task and "2" to indicate arrival at the task location.
- When work is done, the crew sends a text message of "3". In response, the system sends the next task.

  Note: The system sends POUs and breaks to the crew in the same way that it sends activities.
- At the end of the shift, the crew sends a "Logoff" message. Typically, an SMS crew enters completion details at the end of their shift when they return to their office.

Chapter 18

GPS

GPS data is used to display the location of resources on map interfaces and to track crews and routes on the map. This data can be sent from a crew's MDT or directly from an external GPS system embedded in their vehicle.

The MCP sends batches of GPS data to the server and calls the business service, M1-AddGPSData, to upload the data to the database. The business service finds the crew whose active shift is associated with the MDT ID provided as the GPS external identifier and associates the GPS data with the crew.

MDT Type specifies whether the device has GPS capability, the GPS port, how frequently to poll and how frequently to send data to the server. A shift may be configured to allow or disallow sending of GPS data for business reasons. Based on that configuration, a crew shift BO plug-in enables the sending of GPS data.

If GPS data should be sent from a GPS system embedded in a vehicle, the interface to that system should call the same upload service using the XAI Inbound Service "M1-AddGPSCoordinates" providing it with the vehicle's tag as its GPS external identifier. The service then searches for a vehicle with a vehicle tag that matches the GPS external identifier and associates the data with that vehicle.

Chapter 19

Web Services

XAI Inbound Services are used to invoke internal services or functions by an external system.

The application creates a synchronization request to send outbound messages to an external system. Synchronization request records are processed by the Sync Request monitor batch process and it is at that time that the actual outbound message is prepared and sent to the external system.

It important that you schedule the Sync Request monitor batch process to run as frequently as needed to meet your business requirements.

This section provides a high level description of web services and outbound messages provided by the product to support various types of integrations for the following functional areas:

- Appointment Booking
- Activity Maintenance By Host
- Activity Completion To Host
- New Activities to Host
- Contractor Management
- Crews Managed By Host
- Depot Management
- Miscellaneous

Appointment Booking

The host system uses the **Appointment Booking** XAI inbound service to request a list of available appointment time windows for an activity prior to interfacing it. Requests from the host specify the time periods for which the scheduler should return a list of available appointment windows.

The system processes an appointment booking request from the host as follows:

- Using the appointment booking group specified on the request, identify the actual time windows during which appointments can be scheduled. If no appointment booking group is supplied by the host, the system uses the activity type's default appointment booking group.
- Geocode the proposed activity's address.
- Determine the activity service area. This is done by invoking the **Determine Service Area** algorithm associated with the activity business object defined on the proposed activity's type.
- Calculate its arrival time window (ATW) and service time windows (STW) based on the requested Effective Time Windows (ETW) and preferred time windows (PTW). This is

done by invoking the **Update Time Windows** algorithm associated with the activity business object defined on the proposed activity's type.

- If scheduling priority is not provided default it from activity type and business priority number if any.
- Default contractor eligibility and preferences based on activity type and service area.

- The information gathered above is sent to scheduler(s) responsible for appointment booking.
- Appointment information is returned. If there are multiple schedulers handling appointment booking requests, the first response is used.

Activity Maintenance By Host

The host system uses the following XAI inbound services to interface activities to be scheduled, sending updates to activity information as well as inquiring as to the current status of an activity.

Maintain Activity By Host

Use this service to interface a single activity as well as update its information once interfaced.

Update requests to an already completed or canceled activity are not accepted.

By default a new activity or a change to an existing activity is first accepted by the system unconditionally and then later on is sent to the scheduler to confirm a schedule if possible. If a schedule cannot be accommodated by the end of the activity time window and it expires then the system follows the activity expiration rules as configured on the activity type.

A change to an activity, even a dispatched activity, does not recall it from its assigned crew. The system accepts the change but does NOT recall the activity. If the crew eventually does not perform the work then the activity is rescheduled based on updated details.

This is because the system cannot guess which change warrants a recall and which does not. The host system should explicitly cancel can activity to request a recall. Refer to External System Activity Cancellation service for more information on the cancelation process.

The host may indicate that the new activity or changes to an existing activity should only be accepted if can be scheduled as requested. If the host sends the request as a **conditional booking** request the system consults the scheduler real time and only accepts the new or changed activity if the scheduler confirms it can be booked as such.

As mentioned, the system does not recall an activity unless explicitly canceled. If the activity is already dispatched and a change is requested as a conditional booking the system is requested to ensure the "new" booking details can be scheduled as requested if and when the crew returns their assignment and the activity would be rescheduled.

Depending on the type of work requested, it may be important to only accept the "new" booking if the "existing" booking can be guaranteed to be canceled. In other words, only accept the new if it can replace the existing.

Once an activity is dispatched the system can only request a recall but cannot guarantee the work would not be performed as the crew may not be connected to notify the latest status of the activity. If it is important to only accept a new booking as a replacement of an existing booking then the host should also indicate that the request is a **guaranteed rebooking**.

If a guaranteed rebooking is requested the system will error and not accept the change if the activity is dispatched. This is because the system cannot guarantee to the host that the work that was dispatched already to the crew would not be performed.

If it is acceptable by the host that rebooking is not be guaranteed the host system should send in a request to cancel the existing activity and add a new activity with the new details using the Maintain Activities (Job) By Host service. Refer to this service for more information.

Maintain Activities (Job) By Host

Use this service to interface multiple activity as well as update their information once interfaced.

This service is useful when interfacing a set of activities that should be scheduled as a job or a chain or when sending in a rebooking request that cancels one activity and adds another to replace it as a single transaction.

Note that you can still add each activity, even if it is part of a job or a chain dependency, using the single activity service. Using this service though enables the system to validate the set of activities and either accept or reject all in a single transaction.

If conditional booking is requested but it should not be a guaranteed rebooking the host should send the request as a cancelation of the existing activity and a request to create a new activity with the new booking details. The system will confirm the new details with the scheduler and if can be scheduled the system will attempt to recall the activity from its currently assigned crew by calling the External System Activity Cancellation service. Refer to this service for more information.

If the crew has not performed the work and is connected then the recall would be successful and the existing activity would eventually be canceled for which a cancelation message is sent to the host as usual. If however the crew has completed the work the completion information would be sent to the host and it is the host's responsibility to cancel the new activity.

External System Activity Cancellation

Use this service to cancel an activity.

If the activity is not yet dispatched then it is canceled right away and a successful response is sent to the host system.

If the activity is already canceled or completed an error is issued.

Once an activity is dispatched the system can only request a recall but cannot guarantee the work would not be performed as the crew may not be connected to notify the latest status of the activity. If the activity is dispatched the system responds with an indication that it attempts to cancel the activity and has alerted a dispatcher of the situation.

In this situation the system sends a recall message to the crew if they are using a mobile device and raise an alert the crew in case the crew cannot be reached.

If the recall is successful the activity would be canceled and the host would be notified.

If the dispatcher manages to contact the crew and prevent the work then the crew would return their assignment which would also cause the activity to be canceled and notify the host system.

If however the crew has already completed the work the host the completion message is sent to the host as usual.

This means that when an activity is dispatched the immediate response to the host is that cancelation is not confirmed and the host should expect either a cancelation or a completion message from the system when the activity state is finalized.

Retrieve Activity and Assignment Details

Use this service to inquire upon the current status and details of an activity.

Activity Completion To Host

In an integration where activity completion information is entered into the system, by crews or external capacity contractors, the system forwards this information to the host system requesting the work.

The system may also be configured to notify the host system of interim activity status updates. This is not necessary if the host is designed to request a status update on demand using the "Retrieve Activity and Assignment Details" XAI Inbound Service.

All messages to the host are sent to the external system referenced on the activity type.

The system uses the corresponding synchronization request business object for each message type as specified on the M1-ExternalSystemMessages extendable lookup record that matches the host external system code.

The system initiates an **Activity Completion** synchronization request to send activity completion information to the host system when an activity is canceled or completed. If the host has initiated the cancelation or completion then no synchronization request is sent back to the host system.

If enabled on the external system messages extendable lookup, the system initiates an **Activity Status Update** synchronization request to send interim activity status updates once dispatched.

New Activities to Host

Based on your business practices a crew may create activities in the field. A dispatcher may also create an activity for a crew to assist another crew to work an activity.

Any activity created in the system has to be communicated back to the host system associated with the activity type.

The system initiates a **New Activity** synchronization request to send the details of a new activity to the host system when it is created. It uses the corresponding synchronization request business object for this message type as specified on the M1-ExternalSystemMessages extendable lookup record that matches the host external system code.

Contractor Management

The following section describes message exchanges with a contractor's system.

Refer to Integration with Contractor Systems for more information.

Activity to Contractor

All messages to the contractor's system are sent to the external system referenced on the contractor's resource group record.

The system initiates an **Activity to Contractor** synchronization request to send activity details to the contractor's system when an activity is committed to that contractor. It initiates the same type of process when activity details change as well as when the commitment is recalled

When the synchronization request record is processed by the Synch Request monitor it prepares the actual outbound message and sends it to the external system.

The system uses the synchronization request business object specified on the M1-ExternalSystemMessages extendable lookup record that matches the contractor's external system code.

Activity Commitment By Contractor

The contractor system uses this XAI Inbound Service to approve or reject an activity committed to them.

Activity Completion By Contractor

The contractor system uses this XAI Inbound Service to send in completion information for an activity.

Crews Managed by Host

This section is relevant to an integration where crews are managed by the host system and communicate status and completion information directly to the host system. In this type of integration the system is mainly used for planning, scheduling and dispatching purposes but not to capture completion information.

All resource information contributing to establishing mobile workers, crews and shift templates and exceptions are defined in the host system and synchronized with the system. This is then used to generate corresponding entities and shifts needed for scheduling.

The system sends generated shifts to the host system as well as any updates made to shift information.

The host system then notifies the system when shifts start which causes work to be dispatched back to the host system. The system notifies the host system of each dispatched task and in return it is informed of task and shift status changes but not completion information.

All messages to the host system are sent to the external system references on the crew record. The system uses the corresponding synchronization request business object for each message type as specified on the M1-ExternalSystemMessages extendable lookup record that matches the host external system code referenced on the crew record.

Maintain a Common Location

The host system uses this XAI Inbound Service to synchronize common locations defined by the host system.

Maintain Service Area

The host system uses this XAI Inbound Service to synchronize service areas defined by the host system.

Maintain Shift Weekly Template

The host system uses this XAI Inbound Service to synchronize shift weekly templates defined by the host system.

Maintain Work Calendar

The host system uses this XAI Inbound Service to synchronize work calendars defined by the host system.

Process Employee Sync Requests

The host system uses this XAI Inbound Service to synchronize an employee's information as defined by the host system.

Update Shift Status

The host system uses this XAI Inbound Service to notify the system of shift status changes. This service requires a shift ID, the state it changes to as defined by the M1_SHIFT_STATUS_FOR_HOST_FLG lookup, and the time that the status changed.

Update Task Status

The host system uses this XAI Inbound Service to notify the system of task status changes.

Dispatching A Task

The system initiates an **Activity Dispatched** synchronization request to notify the host system that an activity has been dispatched or advance dispatched prior to shift start.

Crew Shift Synchronization

The system initiates a **Crew Shift Synchronization** request to notify the host system of a new shift or a change to an existing shift.

This process relies on the Resource monitor batch process to review and detect shift information changes associated with externally managed crews. It is important that you schedule this batch process to run frequently to meet your business requirements.

Depot Management

Request Depot Cutoff

A depot cutoff request closes all distribution depot runs that are planned to depart from the depot before a cutoff time and instructs the scheduler to build new runs to depart after that cutoff time.

The host system uses this XAI Inbound Service to initiate a depot cutoff process for a specific depot and a cutoff time.

Miscellaneous

Add GPS Coordinates From Host

If crews are using the product's mobile application over devices that support GSP data capture you can configure the devices to send this information to the server.

If however the source of your GPS data is different you may use this XAI Inbound Service to interface GPS data from an external system.

GPS data records have to be associated with either a crew or a vehicle known to the system. Reference a crew name or vehicle tag respectively in the external identifier on each GPS data record.

Geocode an Address

An external system may use this service to geocode an address. The service returns multiple addresses options if more than one is found to match the request.

Appendix A

System Setup Quick Reference Table

This document lists and describes all objects that must be defined as part of the setup process. It identifies the order in which objects should be defined and any prerequisites for setup. Finally, it identifies all objects associated with or referenced by each setup object, thus providing a useful map of the relationship between objects in the system.

This document is divided into the following sections:

- Setup Sequence
- Administration Setup and Maintenance: These setup tasks are typically performed by an admin user and are accessed from the Admin Menu. The objects are defined at implementation and then updated only as needed when system configuration or foundation data changes.
- Resource Setup and Maintenance: These setup tasks are typically performed by a resource planner and are accessed from the Main Menu under the Resource Management submenu. The objects, which include crews, mobile workers, vehicles, and shifts, are defined during initial system setup and updated regularly as new resources are added or changes are made to existing resources.
- Transfer of Goods Setup and Maintenance: These setup tasks are typically performed by an admin user and are accessed from the Admin Menu. The objects are defined during initial system setup and implementation and are updated as new depots are added.
- Contractor Management Setup and Maintenance: These setup tasks are typically performed by an admin user and are accessed from the Admin Menu. The objects are defined during initial system setup and implementation and are updated as new contractors are added.

Note: All basic Framework setup, including system and database setup and any modifications or extensions to base business objects, must have been completed before beginning these setup tasks.

Refer to the Oracle Utilities Application Framework documentation for more information.

Setup Sequence

In the setup tables that follow, the **Sequence** column displays the following codes:

- **L1** = Object has no prerequisites and should be defined before L2-L6 objects.
- **L2** = Object has some L1 prerequisites and should be defined after all L1 objects have been defined and before L3 objects.

- **L3** = Object should be defined after all L1 and L2 objects have been defined.
- **L4 =** Object should be defined after all L1, L2, and L3 objects have been defined.
- **L5** = Object should be defined after all L1, L2, L3, and L4 objects have been defined.
- **L6** = Object should be defined after all other objects have been defined.

Administration Setup and Maintenance

To access the maintenance portals for the objects in this section, do one of the following:

- If you are using functional menus, select **Admin Menu>[***Functional Menu***]>[***object name***]**
- If you are using alphabetical menus, select **Admin Menu**>[*object name*]

The [*Functional Menu*] and [*object name*] are provided below:

Seq	Object	Functional Menu	Description	Prerequisites	Associated with / Referenced by
L1	Alert Type	Service Management	A type of alert (situation requiring user attention or intervention)	None	Alerts, KPIs Global Configuration
L1	Appointment Booking Group	Service Management	Defines valid time windows for appointment booking	None	Activity Types ABG Scheduler Read algorithm
L1	Batch Control		Define a batch control for each scheduler.	None	Schedulers
L1	BI Configuration	BI Configuration	Defines attributes related to creating, maintaining and working with Oracle Utilities Analytics configurations.	None	BI Configuration Integration only
L1	Common Attachments		Create and upload attachments needed to be defined on activity types	None	Activity Types
L1	Country	General	Your organization's country.	None	
L1	Crew Hierarchy	Resource Management	Your organization's logical structure of crews.	See Chapter 10: Configuring Hierarchies	Crews
L1	Crew Type	Resource Management	Type of crew	None	Crews
L1	Currency	Financial	Your organization's native currency.	None	
L1	Dispatcher Type	Resource Management	Type of dispatcher	None	Dispatchers
L1	Display Profile	General	Controls how dates, times, and numbers displayed.	None	
L1	Equipment	Resource Management	Non-human mobile resource capability used by a crew	None	Vehicles, Vehicle Types (assumed equipment), Activity Types (equipment required)
L1	Feature Configuration	System	Used to configure special zones, including the **Gantt** and **Calendar** zones.	None	
L1	Language	General	The language to use for this implementation.	None	

Seq	Object	Functional Menu	Description	Prerequisites	Associated with / Referenced by
L1	MDT Type	Device Management	Defines a type of mobile data terminal, including display options, and synchronization details	None	MDTs, Deployment Types
L1	Parameter Definitions	Scheduler	Parameters used by the scheduler. A complete set of definitions and their default values are shipped with the product. This setup step is only required if you need to override the user minimum/maximum values.	None	Scheduler Configuration
L1	Period of Unavailability (POU) Type	Resource Management	A period of time in which a crew is planned not to perform work, such as while attending a meeting	None	Periods of Unavailability, Period of Unavailability Templates
L1	Priority Profile	Service Management	Allocation factors used to prioritize the scheduling of activities	None	Activity Types
L1	Procedure Type	Resource Management	Type of procedure. Procedures are steps or tasks that must be completed before starting work.	See Chapter 6: Configuring Procedures	Procedures Activity Types
L1	Remark Type	Service Management	Type of activity completion remark	None	Activity Types
L1	Service Area Hierarchy	Service Management	Your organization's logical structure of service areas.	See Chapter 10: Configuring Hierarchies	Service Areas
L1	Shift Type	Resource Management	Type of crew shift	None	Crew Shifts and Crew Shift Templates
L1	Skill	Resource Management	Human skill required to complete certain types of work	None	Mobile Workers, Mobile Worker types (assumed skills), Activity Types (skills required)
L1	Status Reason	System	The reason why an entity transitioned to its current state. This setup step is only required if the base set of status reasons is not adequate for your needs.		Business object's lifecycle
L1	Time Zone	General	Your organization's base time zone.	None	
L1	Timesheet Type	Resource Management	Type of timesheet. Timesheets record time for work completed.	See Chapter 11: Configuring Timesheets	Timesheets
L1	To Do Role	General	Used to associate users with To Do entries.	None	

Seq	Object	Functional Menu	Description	Prerequisites	Associated with / Referenced by
L1	Work Calendar	General	The work calendar for your organization, which identifies your public holidays and is used for scheduling	None	
L1	Work Profile	Service Management	Define operational time windows and other time restrictions regarding when certain types of work can be performed.	See Chapter 4: Work Profile	Activity Types
L1	Extendable Lookup		Various aspects of the application require extendable lookup custom values to be set up for certain functionality to work.		
L2	Cost Profile	Scheduler	Predefined set of parameters and settings used by one or more scheduler configurations.	Parameter Definitions	Scheduler Configuration
L2	Installation Options	System	Control various aspects of the system.	Time Zone, Language, Currency	
L2	Key Performance Indicator	Service Management	Quantifiable measurement that supports dispatcher decision making and in-day exception handling.	Alert Type (if alert-based)	Dispatcher
L2	Location	Resource Management	Predefined physical locations. Use the **Location** maintenance portal, accessible from the **Admin** menu, to configure common locations.	Time Zone Country	Crew Shifts, Crew Shift Templates (as logon/logoff locations), POUs (as event locations)
L2	Master Configuration	System	Enables an implementation to capture various types of information in the system. For ORS, edit the Global Configuration record to suit your needs.	Alert Type, Task Type, Timesheet Type	
L2	Mobile Worker Type	Resource Management	Type of mobile worker	Skills	Mobile Workers
L2	Scheduler Configuration	Scheduler	Predefined set of parameters and settings used by one or more schedulers.	Parameters (cost control)	Scheduler

Seq	Object	Functional Menu	Description	Prerequisites	Associated with / Referenced by
L2	Service Area	Service Management	A logical boundary of an organization's territory that may or may not be based on geography	Service Area Hierarchy	Mobile Workers, Crew Shifts and Crew Shift Templates, Activities, Periods of Unavailability (POUs), Dispatch Areas, Shift Cost Profiles, Scheduler Areas Override Time Windows
L2	Task Type: Activity Type	Service Management	A type of task performed at a particular geographic location.	Priority Profile, Appointment Booking Profile, Skill, Equipment, Remark Type, To Do Type and To Do Roles	Activities (Activity tasks), Service Classes
L2	Task Type: Break Type	Service Management	A type of break	Timesheet Time Type (extendable lookup)	Crew Shift Templates, Break tasks
L2	Task Type: Non-Productive Task Type	Resource Management	A type of a non productive time task a crew is planned not to perform shift related work, such as cleaning or loading vehicles.	Timesheet Time Type (extendable lookup)	Non-Productive Tasks
L2	Task Type: POU Task Type	Service Management	A type of period of time unavailability (POU) task	Timesheet Time Type (extendable lookup)	POU Tasks, Global Configuration
L2	User	Security	Defines a user's user groups, data access roles, portal preferences, default values, and To Do roles	Language, Display Profile, To Do Roles	
L2	Vehicle Type	Resource Management	Type of vehicle	Equipment	Vehicles
L3	Mailing Lists	Mail Management	A list of users for mail distribution.	User	
L3	MDT	Device Management	Defines attributes of this mobile device terminal, including registration/tag/ serial number, system logging data, etc.	MDT Type	Crew Shift
L3	Scheduler Area	Scheduler	Predefined set of service areas and optimization information.	Service Area	Scheduler

Seq	Object	Functional Menu	Description	Prerequisites	Associated with / Referenced by
L3	Service Class	Service Management	Broad categorization of activity types, can be used to restrict a crew shift to certain activity types.	Activity Type	Shift cost profile, Crew Shifts and Crew Shift Templates, Dispatch Area, Activity Type
L3	User Group	Security	A group of users who have the same degree of security access	User	
L4	Deployment Part	Device Management	A collection of deployment items, such as business objects, UI maps, scripts, administrative data used on the device, etc.	Any referenced entities	Deployment Type
L4	Dispatch Area	Service Management	Set of service areas and service classes to be monitored	Service Areas, Service Classes	Dispatchers, Dispatch Shift
L4	Mail Templates	Mail Management	Predefined mail messages.	Mailing Lists if you are using them on templates	
L4	Scheduler	Scheduler	An individual instance of the ORS scheduler.	Scheduler Configuration Scheduler Area Batch Control	
L4	Shift Cost Profile	Resource Management	Set of shift-based cost factors used as multipliers against global costs	Service Areas, Service Classes	Crew Shifts and Crew Shift Templates
L5	Deployment Type	Device Management	Defines attributes of an application to be deployed, including authorized user groups, supported MDT types, etc.	Deployment Part, Message Category, MDT Type, User Group	Deployment
L6	Deployment	Device Management	A specific cut or build of a deployment type for a specific language.	Deployment Type	

Resource Setup and Maintenance

To access the maintenance portals for the following objects, select **Main Menu**>**Resource Management>**[*object name*].

Seq	Object	Description	Prerequisite	Associated with / Referenced By
L1	Alert Area	Defines geographic areas that can be displayed on map and that trigger an alert when GPS enabled resources enter or leave designated areas.	Service Class Vehicle Type	Alert
L3	Mobile Worker	An individual (human resource) performing work as part of a crew	Mobile Worker Type, User (System User ID), Country, Skills, Service Areas If using contractors, Resource Group	Crew Shift
L3	Vehicle	A non-human resource used by a crew	Vehicle Type Equipment If using contractors, Resource Group	Crew Shift
L4	Crew	Uniquely named group of resources (mobile workers and vehicles) scheduled to perform work	Crew Type If using contractors, Resource Group	Shift Weekly Templates, Crew Shifts, POUs, POU templates
L5	Dispatcher	An individual responsible for monitoring and managing day-to-day field operations	User, Dispatcher Type, Dispatch Area, Key Performance Indicators	Dispatcher Shift
L5	Period of Unavailability	A period of time in which a crew is planned not to perform work, such as while attending a meeting	Country, Time zone, Work Calendar, Location, POU Type. Resources (Crews, Mobile Workers, or Vehicles attending POU), Service Areas	POU Task
L5	Shift Weekly Templates	A cyclical weekly pattern made up of Crew Shift Templates	Crew, Work Calendar, Time zone (system setup), Service Classes, Break Types, Mobile Workers, Vehicles, Service Areas, Location (for logon/ logoff), Shift Cost Profile, Shift Type	Crew Shift

Transfer of Goods Setup and Maintenance

To access the maintenance portals for the objects in this section, do one of the following:

- If you are using functional menus, select **Admin Menu>[***Functional Menu***]>[***object name***]**
- If you are using alphabetical menus, select **Admin Menu**>[*object name*]

The [*Functional Menu*] and [*object name*] are provided in the appropriate columns in the following table.

Seq	Object	Functional Menu	Description	Prerequisites	Associated with / Referenced by
L1	Load Capacity Type Look Up fields		Enable relevant goods related capacity types by populating their override description	Timesheet Time Type (extendable lookup)	
L2	Task Type: Depot Task Type	Service Management			Depot Master Configuration
L3	Depot Profile	Resource Management	Establish common settings for site delay, access time windows and service time windows across multiple depots.	Depot Task Type	Depot
L4	Depot	Resource Management		Depot Profile, Depot Task Type	

Contractor Management Setup and Maintenance

To access the maintenance portals for the objects in this section, do one of the following:

- If you are using functional menus, select **Admin Menu>[***Functional Menu***]>[***object name***]**
- If you are using alphabetical menus, select **Admin Menu**>[*object name*]

The [*Functional Menu*] and [*object name*] are provided in the appropriate columns in the following table.

Seq	Object	Functional Menu	Description	Prerequisites	Associated with / Referenced by
L1	Capacity Type	Resource Management	Define capacities of a certain type.		Capacities
L1	Outbound Message Type		Define the structure of outbound messages to the contractor's system. Only applicable if integrated. Refer to the integration section.		External System Schema Constants Configuration
L1	Resource Group Type	Resource Management	Define classifications for various types of contractors your company does business with.		Resource Groups
L2	External System		Defines outbound message XAI configuration with an external system	Outbound Message Type	Resource Group

Seq	Object	Functional Menu	Description	Prerequisites	Associated with / Referenced by
L2	Resource Group	Resource Management	Define information about contractors doing work for your organization.	Resource Group Type Service Area External System	Contractor Eligibility User Mobile Worker, Vehicle, Crew Capacity Weekly Template
L2	Schema Constants setup		Associate the "Activity To Contractor" message type constant with the corresponding outbound message type.	Outbound Message Type	
L3	Capacity Weekly Template	Resource Management	Define weekly work templates for a specific contractor.	Capacity Capacity Type Service Area Service Class Activity Type	
L3	External System Messages - Extendable Lookup		Defines messages applicable for an external system.	External System	

Appendix B

Algorithm Entities

This section lists and describes Oracle Real-Time Scheduler specific algorithm entities, also referred to as plug-in spots.

> **Note**: For instructions on creating custom algorithms, see the Framework documentation. To view detailed information about specific algorithms provided with the base system, use the Application Viewer (also described in the Framework documentation). The Application Viewer provides information about the base logic, inputs, and outputs of each algorithm entity.

Navigate to **Admin** > **Algorithm Type** and search for the **Algorithm Entity** to review these entities. Refer to the detailed description within the application for more information.

Algorithm entities are grouped by type:

- Installation Options
- Activity
- Task
- POU
- Shift and Contractor Capacity
- Scheduler
- Common Dispatching
- Alerts
- KPI
- Zone Contents
- Mobile Application

The **Optional/Required** column in the following tables indicates whether or not the plug-in is required for the system to operate. If Required, the system assumes that a plug-in exists and will generate an error if it does not.

Installation Options

Algorithm Entity (System Event)	Optional / Required	Description
Address Geocoding	Required	Determines the geocoordinates of an address. An algorithm of this type is used to convert a standard address into a geocode longitude/latitude pair. The algorithm is exposed as a business service and called whenever an entity associated with an address needs to be geocoded. The minimum quality level for address geocoding is configurable. By default, the base system requires an exact match, which is the maximum quality level. You can control the quality level of geocoding for any entity that has an address by configuring a custom algorithm on the entity that calls this service with a different minimum quality level, based on your business needs.
Address Information	Required	Formats address information based on standard address constituents and requested format type. The base product supports a single line format as well as a postal format.
Determine Connected Device	Optional	Only required if your organization uses the connected MCP feature. The mobile application may be accessed in a connected mode in which neither data nor mobile application reside on the mobile device. They reside on the server and the user must be connected to the server at all times using their browser to access the application. In this mode the actual physical is irrelevant and therefore registering it as a specific mobile data terminal (MDT) in the system is not needed. However, the mobile application framework still relies on a unique MDT identifier to mark the user's data on the server. An algorithm of this type determines and assigns an MDT record to a user when they log in to the mobile application in a connected mode.
Remote Script Invocation	Optional	Processes a remote message sent from a mobile application to the main application. Only required if your organization uses SMS messaging and mobile devices.
SMS Receive	Optional	Used to process an incoming SMS message. Only required if your organization uses SMS messaging and mobile devices.
SMS Send	Optional	Used to send an SMS message. Only required if your organization uses SMS messaging and mobile devices.

Activity

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
Contractor Eligibility	Optional	An algorithm of this type determines an activity's preferences with respect to contractor scheduling. If the activity is eligible for contracting then it determines whether capacity contractors should be preferred over internal company and contractor shifts. It may also provide an optional ranked list of these resource groups, i.e. the company and the contractor companies eligible to work in the service area, in order of preference. The algorithm is called by a monitor rule on the activity business object "Being Scheduled" state. The monitor rule then capture these preferences onto the activity record so they can be forwarded to the scheduler. The algorithm is also called from the Appointment Booking request process provided with the proposed activity information.	Defined on the activity business object.
Determine Task Procedures	Optional	Retrieves a list of required procedure types for the task. It looks for procedure types associated with the assignment's activity type. If the passed in mode is Create, then a procedure will be created for each required procedure type.	Defined on the activity business object.
Exclude Contractor	Optional	Excludes the activity's currently assigned contractor from performing it and reschedules it as needed.	Defined on the activity business object.
Fix to Crew	Optional	Fixes an activity to a crew so it is assigned only to that crew's shifts. It is also used to reset the fix to a crew.	Defined on the activity business object.
Prepare Activity Completion Details		Prepares completion information common to any type of activity. If the activity is a complex activity then it also summarized crew time information from all completed crew visits.	Defined on the activity business object.
Activity Expiration	Optional	Handles the situation of an expired activity. The algorithm is called periodically by the activity batch monitor process once the activity has been expired letting it either extend the activity, cancel it, raise a to do entry for a user to take action or any other business rule needed by your organization.	Defined on the activity business object.
Determine Service Area	Optional	Determines an activity's service area if not provided by the host system. It may use any detail associated with the activity to do so. For example, the entity may use the activity's postal code. The algorithm is called when a new activity is created as well as when a proposed activity is sent in on an appointment booking request. If you are using the base algorithm, either define the geographic boundaries on each service area or set up the "Postal Code to Service Area Mapping" extendable lookup to associate your postal codes with a corresponding service area.	Defined on the activity business object.
Set Alternate ID	Required	Responsible for assigning a unique alternate activity ID that is relatively easier to handle and shorter in length than the system-generated prime key. This is mainly used by dispatchers to radio-dispatch an activity to a crew when automatic synchronization with their mobile device is not possible.	Defined on the activity business object.

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
Update Time Windows	Required	Responsible for calculating an activity's arrival and service time windows. This algorithm is called when an update is made to an activity or an appointment booking request is received.	Defined on the activity business object.

Task

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
Task Detailed Information	Required	Prepares various formats of detailed task information, based on where the data is to be displayed. For example, it formats a task's detailed information as displayed at the top of the context menu when a user right-clicks on a task in the **Scheduling Gantt** zone. It also formats the data to be displayed when a users hovers over a task-related point on the **CDI Map**.	Referenced on the task's business object.
Task Synchronization	Required	Used to send current task details to the assigned crew. This is called when the task is first dispatched and whenever dispatched information is changed. The algorithm is also called to prepare the preview information for a task when it is previewed before the shift starts. It may also be called to resend a task to the crew if the original mobile device breaks.	Defined on the task business object.

POU

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
Generate POUs	Required	Generates periods of unavailability based on a template.	Defined on the POU template business object.

Shift and Contractor Capacity

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
Generate Crew Shifts	Required	Generates shifts for a crew based on a template.	Defined on the shift template business object.
Prepare Shift Data	Optional	Prepares and adjusts shift details to be used for creating shifts based on details configured on the crew record.	Defined on the crew business object and is called from the shift generation process to further apply crew details on top of a shift template before it is used to create shifts. It is also called when an ad-hoc shift is added online.
Crew Shift Detailed Information	Required	Prepares various formats of detailed shift information, based on where the data is to be displayed. For example, it formats a shift's detailed information as displayed at the top of the context menu when a user right-clicks on a shift in the **Scheduling Gantt** zone. It also formats the data to be displayed when a users hovers over a shift-related point on the **CDI Map**.	Referenced on the shift's business object.
Synchronize Shift	Required	Responsible for sending current shift details to the crew. This is called once when the crew requests the shift at logon time and on certain changes after that.	Defined on the crew shift business object.
Determine Shift Procedures	Optional	Retrieves a list of required procedure types for the task. It looks for procedure types associated with the assignment's activity type. If the passed in mode is Create, then a procedure will be created for each required procedure type.	Defined on the crew shift business object.

Scheduler

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
ABG Scheduler Read	Required	Responsible for obtaining appointment booking group (ABG) details relevant to the scheduler. The schedule manager invokes this algorithm whenever a new or changed appointment booking group needs to be communicated to the scheduler.	Defined on the ABG business object.
Depot Scheduler Read	Required (Depot only)	Responsible for obtaining scheduling details of a depot associated with any of the scheduler's registered tasks and shifts. The algorithm is called when the depot is first time interfaced to the scheduler and upon subsequent changes made to its information. A depot is sent along with its access and service time windows for a requested time range. Time window calculation takes into account holidays as well as explicit overrides canceling or extending normal hours of operation.	Defined on the depot business object.
Scheduler Configuration Read	Required	The schedule manager invokes this type of algorithm to communicate new or changed scheduler configuration details to the scheduler.	Defined on the scheduler configuration business object.
Shift Scheduler Read	Required	Responsible for obtaining shift / contractor capacity details relevant to the scheduler. The schedule manager invokes this algorithm whenever a new or changed shift needs to be communicated to the scheduler.	Defined on the crew shift business object.
Task Post Dispatch	Required	Responsible for executing the necessary business rules and processing associated with a schedule update of a dispatched task. The schedule manager calls it when a task reaches the dispatched scheduler state and on every schedule update to it after that.	Defined on all task business objects except for assignment business object.
Task Scheduler Read	Required	Responsible for obtaining task details relevant to the scheduler. The schedule manager invokes this algorithm whenever a new or changed task needs to be communicated to the scheduler.	Defined on all task business objects except for assignment business object.
Scheduler Performance Level of Service	Optional	Assesses the level of service of the scheduler.	Defined on the batch control. Associated with the scheduler record.

Common Dispatching

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
Allocate to Shift	Required	Performs business rules necessary to allocate an activity to a particular crew's shift.	Defined on the activity business object.
Cancel Activity	Required	Performs business rules necessary to cancel an activity.	Defined on the activity business object.
Defer Activity	Required	Performs business rules necessary to defer scheduling of an activity to a specific date.	Defined on the activity business object.
Lock Depot Run	Required (Depot only)	Locks a depot related activity to its currently assigned depot task. The latter represents the scheduled run to the depot to pickup or drop off goods.	Defined on the depot task business object.
Unassign	Required	Unassigns an activity from its currently assigned crew if any and set to be rescheduled. If it is currently allocated to a specific crew, this also de-allocates the activity from that crew, making it eligible for all qualifying crews.	Defined on the activity business object.

Alerts

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
Create Alert	Optional	Responsible for raising new alerts of a given type.	This algorithm is defined on the alert type record and is called by a monitor algorithm on the alert type business object.
Evaluate Alert	Optional	Responsible for evaluating an existing alert to determine whether it can be closed.	This algorithm is defined on the alert type record and is called by an alert business object monitor algorithm executed when the alert monitor background process runs or when updates are made to the activity/crew shift associated to an alert.

KPI

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
KPI Calculation	Required	Calculates details related to a KPI for a given dispatcher based on a requested mode of operation. By default, this algorithm calculates the KPI measurement value and formats its summary description for display. It may also be called to provide detailed, actionable lists of the actual activities and shifts contributing to the measurement.	Defined on the KPI record.

Zone Contents

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
CDI Gantt Data	Required	Prepares the data to be displayed on the common dispatcher interface **Gantt** zone.	Referenced on the **Scheduling Gantt** zone as a zone parameter.
Calendar	Required	Supports a special kind of zone that displays a calendar populated with objects for display. An algorithm of this type prepares the data to be displayed on an entity's calendar zone.	Referenced on the **Calendar** zone as a zone parameter.

Mobile Application

The following algorithms are only applicable to the mobile application.

Note: These MCP Business Object algorithms are modeled after the server concept of the corresponding algorithms, but apply only to the mobile device and have a different hard parameter interface.

Unlike the server framework, when these MCP algorithms are called the entity in the local database still reflects the unchanged version, while the changed entity is provided in memory as an input hard parameter. Further changes to the entity should be made to the input copy, not directly

to the database. Once all algorithms are processed, the mobile framework persists the changed version to the database.

Algorithm Entity (System Event)	Optional / Required	Description	Configuration
Capture Content	Optional	Implements a specific type of digital content capture on a mobile device for a given entity, for example capture sound or picture related to an activity. The algorithm posts the captured file on a designated file location on the device. It may also be used to attach an existing file on the mobile device to the activity. When synchronized to the server the file is stored as an attachment to the related entity. A business service is provided to invoke such algorithms as needed by the mobile application.	The MDT type has to be configured with such an algorithm as a capability.
General Event	Optional	An algorithm of this type could be used to implement a generic mobile device capability. It accepts a data area in raw format as well as the capability type in context. A business service is provided to invoke such algorithms as needed by the mobile application.	The MDT type has to be configured with such an algorithm as a capability.
Map Layers	Optional	Determines the applicable list of map viewer layers to be displayed when the geographic map is launched on the mobile application for a specific shift. If you are using the base algorithm, set up the "Mobile Device Map Viewer Themes" extendable lookup to list your map viewer theme layers.	Configured on the crew shift business object.
Mapping	Optional	Launches a geographic map on a mobile device. The algorithm is called when the user clicks on the map icon. The icon is available for devices that were configured to support mapping capability.	The MDT type has to be configured with such an algorithm as a capability.
MCP Enter	Optional	Applies business rules when a business object instance enters a given state on the mobile application.	
MCP Post Processing	Optional	Used to perform additional business logic after a business object instance has been processed on the mobile application.	

Appendix C

Batch Controls

This chapter provides descriptions of the base package batch processes and describes the delivered tools that can be used to help manage batch processes, including:

- Batch Control Global View
- Timed Processes
- Commit Frequency
- Monitors
- Schedulers
- BI Batch Controls
- Base Package Batch Controls to Run Periodically
- Base Package Batch Controls to Run Infrequently

Use the **Batch Control** portal and the A**pplication Viewer** for more details about batch processes.

Batch Control Global View

The **Batch Control Global View** portal provides an "at a glance" view of any processes controlled by batch including monitors, schedulers and other batch controls.

Monitor batch processes as well as scheduler batch processes are configured to use a Level of Service algorithm defined on their batch control record to assess how well the process currently runs.

When these algorithms are used with batch processes, they can answer the question of whether or not the batch process is running as expected. The diagnostics provide additional supporting information so that users can troubleshoot and otherwise monitor the processes. For example, slow running processes and job failures are highlighted. Refer to the description of each algorithm for more information.

When this functionality is properly configured, diagnostic information is available both in the **Batch Control Global View** portal and in the **Batch Control** portal for the specific batch process.

> **Note**: Your implementation can use a custom level of service algorithm based on your business rules.

Timed Processes

Some batch processes are required to run periodically and others on demand. Periodic processes may need to be run very frequently like once a minute while others may be run once a day etc.

If your organization does not use a third party software to trigger batches processes periodically you can configure the batch control as a timed batch process and specify the appropriate frequency it should be perform.

Refer to the description of each batch control for more information.

Commit Frequency

To improve overall batch processing performance, Oracle Utilities recommends setting the commit interval for all batch controls to "1". This can be set in the following ways:

- Using Online Batch Job Submission
  When submitting a batch job online via the **Batch Job Submission** portal, you can set the commit interval by specifying the appropriate value in the **Override Nbr Records to Commit** field.

- Using the Submitjob utility
  When submitting a batch job via the Submitjob[.sh] utility, you can set the commit interval by specifying the appropriate value in the **–f <record count>** switch.

- Setting the Default Commit Interval
  To set a default commit interval, specify the appropriate value for the **com.splwg.batch.submitter.maximumCommitRecord**s parameter in the **submitbatch.properties** configuration file.

Refer to the Batch Server Administration Guide for more information.

Monitors

Monitors batch processes invoke business object rules to determine when an event has taken place within the system and trigger another event in response.

As a general rule, all monitor algorithms are executed the first time an entity enters a given state. After that event, the system relies upon the monitor batch process configuration to periodically call them thereafter.
Most monitor batch process needs to be executed periodically where each requires a different frequency. Refer to each batch process for more information.

All monitor processes may be run multi threaded.

> **Warning!** All monitor processes must be set up with commit frequency set to 1.

Schedulers

Each scheduler is managed by its own scheduler manager batch process. The batch control of a scheduler is referenced on its scheduler record.

Scheduler manager batch processes must be run continuously for the schedulers to work properly.

If your organization does not use a third party software to trigger and monitor batches processes periodically you may configure them to be timed with a 60 seconds interval.

You may also configure scheduler manager processes to run on designated thread pool workers for load balancing purposes.

BI Batch Controls

The base product provides several batch controls that control extract and load functions for the integration with Business Intelligence. These need to be run periodically if needed. Refer to Oracle Utilities Analytics documentation for more information.

Base Package Batch Controls to Run Periodically

This section lists the batch controls that are provided in the base package and that must be run on a regular basis. You are free to configure these batch controls in any way that suits your business practices, however the table provides a suggested timer interval for each type of batch control.

Batch	Description	Recommended Timer Interval
M1-CLMDT	Stale MDT and RSI Cleanup	Once a day.
M1-DPUTD	Deployment Evaluation and Purge	Once a day.
*your scheduler batch controls	Scheduler Manager Batch Job	Run constantly.
F1-SYNRQ	Sync Request Monitor Process	Every 5 - 10 minutes. This is a FW owned batch process but ORS uses this entity to send messages to the host system so it needs to run as often as these messages need to be sent.
M1-ALMTR	Alert Monitor	60 sec - This monitor should run very frequently to evaluate existing alerts and recycle them in a timely manner.
M1-ATMTR	Alert Type Monitor	120 sec - This monitor should run very frequently so it can raise alerts as soon as they occur and bring them to a user's attention.
M1-CRSHF	Crew Shift Monitor	5 - 10 minutes. Base shift monitor rules are typically used to send estimated arrival time updates to crews for their schedule, so they should be frequent, but it is not necessary for them to be by the minute.
M1-DEPOT	Depot Monitor	Once a day.
M1-DSMTR	Dispatcher Shift Monitor	Frequently (every 60 seconds). This control refreshes the shifts associated with dispatcher shifts. It is important to run this process very frequently as it ensures the dispatcher's scope of shifts is up to date.

Batch	Description	Recommended Timer Interval
M1-POU	Period of Unavailability Monitor	Once a day to generate POUs based on template.
M1-RESRC	Resource Monitor	Should be run frequently if resources are managed by an external system otherwise unless you added custom rules it should not be run.
M1-RMMSG	Remote Message Monitor	Every 5 - 10 minutes.
M1-RSIBP	Remote Script Invocation Manager	Must run constantly.
M1-SCHED	Scheduler Monitor	Every few hours.
M1-SHWKT	Shift Weekly Template Monitor	Once a day to generate shifts.
M1-TSKTR	Task Monitor	Every 30 or at least once an hour.

Base Package Batch Controls to Run Infrequently

This section lists the batch controls that are provided in the base package and do not need to be run often or at all.

Batch	Description	Recommended Timer Interval
M1-PRMTR	Procedure Monitor	There are no base monitor rules for these batch controls, so unless you add custom rules, these should not be run.
M1-TTWTR	Override Time Window Monitor	
M1-DPTTW	Depot Time Window Monitor	
M1-MAIL	Mail Monitor	
M1-SCHCF	Scheduler Configuration Monitor	
M1-DPLOY	Create Deployment	This should only be run when a deployment item has changed and a new deployment version needs to be created.

Appendix D

Configuration Migration Assistant (CMA)

The Configuration Migration Assistant (CMA) provides customers with a flexible, extensible facility for migrating their configuration data from one environment to another e.g., from a development environment to a production environment. Data is exported from the source system to a file. The file can then be checked in to a version control system for reuse, or can be immediately imported into the target system and applied.

This appendix describes how the Configuration Migration Assistant can be used with Oracle Utilities Mobile Workforce Management and Oracle Real-Time Scheduler.

Read and review the **Configuration Migration Assistant** section in the *Oracle Utilities Application Framework Administration Guide* before attempting to use this functionality.

At a high level, migrating data involves the following steps:

1. Create one or more structured **Migration Plans** and specify one or more business objects to migrate.
2. Create a **Migration Request** to specify one or more Migration Plans for export and the specific objects to export using that plan.
3. Create a **Migration Data Set Export** object to register the intent to export specified requests.
4. Create a **Migration Data Set Import** object to register the intent to import completed exports.
5. Review the changes detected by the import and compare process.
6. Apply approved changes.

The following sections provide a reference to the information needed to configure objects and criteria for migration, including the following:

Migration Requests

Migration Requests are unordered lists of Migration Plans that are to be migrated together. Algorithms, SQL statements, or specific keys are used to specify the set of objects you want to export. When complete, the request describes the complete data set that is extracted to the migration export file when the request is executed.

The base package migration requests provided can be used as a basis for custom requests. Use the following procedure to access the base package migration requests:

1. Navigate to **Admin Menu** > **Migration** > **Migration Request**.
2. Enter "M1" in the **Migration Reques**t field.
3. Click **Refresh**.
4. The **Migration Request Search** results list displays a list of the base package migration requests:
5. Click the **Description** link in the search results list for the migration request you wish to work with.

Base Package Migration Requests

This section outlines the details of the base package migration requests. Use the **Migration Request** portal to view additional details about each migration request.

Admin Data

- **Migration Request**: M1-Admin
- **Description**: Admin Data
- **Detailed Description**: This migration request migrates user defined custom control data entities.
- **Migration Plans**: This migration request includes the following base package migration plans.
    - Alert Type (M1-AlertTypePhysicalBO)
    - Capability Type (M1-CapabilityTypePhysicalBO)
    - Crew Shift Type (M1-CrewShiftTypePhysicalBO)
    - Dispatch Area (M1-DispatchAreaPhysicalBO)
    - Hierarchy (M1-HierarchyPhysicalBO)
    - KPI (M1-KPI)
    - Location (M1-LocationPhysicalBO)
    - Mailing List (M1-MailingListPhysicalBO)
    - POU Type (M1-PouTypePhysicalBO)
    - Priority Profile (M1-PriorityProfilePhysicalBO)
    - Procedure Type (M1-ProcedureTypePhysicalBo)
    - Remark Type (M1-RemarkTypePhysicalBO)
    - Resource Type (M1-ResourceTypePhysicalBO)
    - Service Area (M1-ServiceAreaPhysicalBO)
    - Service Class (M1-ServiceClassPhysicalBO)
    - Shift Cost Profile (M1-ShiftCostProfilePhysicalBO)

- Task Type (M1-TaskTypePhysicalBO)
- Time Profile (M1-ApptBookGrpPhysicalBO)
- Timesheet Type (M1-TimesheetTypPhysicalBO)
- Geographic Area (M1-GeographicAreaPhysicalBO)

Deployment Configuration

- **Migration Request**: M1-DeploymentSetup
- **Description**: Deployment Configuration
- **Detailed Description**: This migration request migrates deployment configuration entities.
- **Migration Plans**: This migration request includes the following base package migration plans.
  - Deployment Part (M1-DeploymentPartPhysicalBO)
  - Deployment Type (M1-DeploymentTypePhysicalBO)
  - MDT Type (M1-MDTTypePhysicalBO)

Depots

- **Migration Request**: M1-Depots
- **Description**: Depots
- **Detailed Description**: This request migrates depots.
- **Migration Plans**: This migration request includes the following base package migration plans.
  - Depot (M1-DepotPhysicalBO)

Geographic Areas

- **Migration Request**: M1-GeographicAreas
- **Description**: Geographic Areas
- **Detailed Description**: This migration request migrates geographic areas. If the geographic area is an alert area then it can only be exported if the area does not reference excluded resources.
  These entities also exist as part of the admin setup migration request.
- **Migration Plans**: This migration request includes the following base package migration plan:
  - Geographic Area (M1-GeographicAreaPhysicalBO)

Scheduler Control Data

- **Migration Request**: M1-SchedulerSetup
- **Description**: Scheduler Control Data
- **Detailed Description**: This migration request migrates scheduler related entities.
- **Migration Plans**: This migration request includes the following base package migration plans.
  - Scheduler (M1-SchedulerPhysicalBO)
  - Scheduler Area (M1-SchedulerAreaPhysicalBO)
  - Scheduler Configuration (M1-SchedConfPhysicalBO)
  - Service Area (M1-ServiceAreaPhysicalBO)

Task Types

- **Migration Request**: M1-TaskTypes
- **Description**: Task Types
- **Detailed Description**: This migration request migrates all task types and service classes. These entities also exist as part of the admin setup migration request.
- **Migration Plans**: This migration request includes the following base package migration plans.
  - Task Type (M1-TaskTypePhysicalBO)
  - Service Class (M1-ServiceClassPhysicalBO)
  - POU Type (M1-PouTypePhysicalBO)

Migration Plans

Migration Plans comprise a group or groups of objects to migrate as a set of related objects. Essentially migration plans sets of instructions describing how the data to be exported is structured. Migration plans allow objects to be migrated together as a logical unit to ensure consistency and completeness.

The base package contains migration plans which can be used as a basis for custom plans. Use the following procedure to access the base package migration plans:

1. Navigate to **Admin Menu** > **Migration** > **Migration Plan**.
2. Enter "M1" in the Migration Plan field.
3. Click **Refresh**.
4. The **Migration Plan Search** results list displays a list of the base package migration plans:
5. Click the **Description** link in the search results list for the migration plan you wish to work with.

Use the **Migration Plan** portal to view additional details about these migration plans.

Note that base package migration plans cannot be altered. To extend an existing base package migration plan, the base package plan must be duplicated and then altered as needed. The duplicate migration plan can then need to be added to a custom migration request.

Wholesale and Piecemeal Migrations

There are two general types of migrations used with the Configuration Migration Assistant: wholesale migrations and piecemeal migrations.

Wholesale Migrations

Wholesale migrations are used when migrating all the configuration and/or admin data from one environment to another. For example, a wholesale migration might be used when migrating admin data from a development or test environment to a production environment.

Wholesale migration plans contain Primary Instruction as well as Subordinate Instructions that involve "CLOB-embedded links" (if any exist). Subordinate Instructions for "Standard Constraints" may be included but are deliberately omitted because they are extraneous, bulky and redundant for wholesale migrations. Wholesale migration plans are included in the base package "Admin Data" migration request (M1-Admin) and are provided for use with wholesale migrations.

Executing Wholesale Migrations

This section provides a high-level overview of the steps involved when executing a wholesale migration.

1. Process the "F1-Languages" (Languages) migration request.
2. Process the "F1-SchemaAdmin" (FW Foundation) migration request (This request contains migration plans for Field, Lookup, Char Type, Currency Code and FK Ref).
3. Process the F1-FrameworkAdmin (Framework Admin) migration request This request contains migration plans for Business Objects, Algorithms, and Extendable Lookups, and other FW-based admin objects.
4. Process the "M1-Admin" migration request. This includes several of the base package M1-owned migration plans.
5. Process any of the other delivered M1-owned migration requests as needed. This includes the following:
   - Deployment Configuration (M1-DeploymentSetup)
   - Depots (M1-Depots)
   - Scheduler Control Data (M1-SchedulerSetup)
6. Process any of the other delivered framework-based (F1-owned) migration requests as needed.

All migration requests can be exported at the same time. When importing, Oracle recommends importing, reviewing and applying an entire file/data set before moving on to the next one. If there are objects included in more than one file (which can happen), then two sets of "inserts" will be generated, and only the first will succeed. The second will cause an insert error on that object, and the transaction would be put into "Applied with Error" status. Waiting to perform the import of a second file until after the first is applied means the second dataset will not generate any SQL (since the object is already inserted). When importing all files at once then trying to apply them all, duplicated objects will have to be identified as errors and be marked as "Rejected" before the transaction can be applied.

Piecemeal Migrations

Piecemeal migrations are used when migrating a small portion (or piece) of configuration and/or admin data from one environment to another. For example, a piecemeal migration might be used when migrating alert types and deployment types from a development or test environment to a production environment.

Piecemeal (or non-wholesale) migration plans contain both Primary Instruction as well as all Subordinate Instructions. Subordinate Instructions include "CLOB-embedded links" (if any exist) as well as "Standard Constraints".

Executing Piecemeal Migrations

This section provides a high-level overview of the steps involved when executing a piecemeal migration.

1. Create one or more piecemeal migration plans as needed based on the specific configuration/admin data to be migrated.
2. Create a custom migration request that contains the migration plans created in step 1, and specifies the keys of the objects to be migrated. Keys can be specified using any of the selection types - SQL, algorithm or specific key values. Be sure to include all primary key values for the objects.
3. Process the custom migration request (export/import).

When importing piecemeal migrations, Oracle recommends importing, reviewing and applying an entire file/data set before moving on to the next one (similar to the steps recommended for wholesale migrations).

Base Package Piecemeal Migration Requests

The following base package migration requests are provided to support piecemeal migrations of specific types of configuration data.

- Deployment Configuration (M1-DeploymentSetup)
- Depots (M1-Depots)
- Geographic Areas (M1-GeographicAreas)
- Scheduler Control Data (M1-SchedulerSetup
- Task Types (M1-TaskTypes)

Data that Cannot be Migrated Using Configuration Migration Assistant

This section provides details regarding data that cannot be migrated using the Configuration Migration Assistant.

Key Type

Only data with specific types of keys can be migrated using the Configuration Migration Assistant. To be included in a base package migration, the "Primary" table for the object's maintenance object main must meet the following criteria:

- KEY_TYPE_FLG = 'USR' (User-defined).

Note: KEY_TYPE_FLG is a column in CI_MD_TBL.

Because they all use system-generated keys, the following maintenance objects are not supported by Configuration Migration Assistant:

- Shift Weekly Template
- POU Template
- Mail Template

Data based on these objects must be migrated manually between environments.

Links to System Generated IDs

Data with links to system-generated IDs are not supported in the Configuration Migration Assistant.

- Alert areas that reference excluded resources cannot be migrated because IDs for resources are system generated.

Configuration Migration Assistant does not currently support migration of objects with system-generated IDs. As of this release, alert areas that reference excluded resources must be migrated manually.

Appendix E

Mobile Communication Platform Custom Business Services Development Setup Guide

This document describes how customers can create their own custom business services using the MCP Business Service API. This includes information on development environment setup, API usage, implementation guidelines, deployment, and unit testing.

This section includes the following:

Set Up the Eclipse Environment

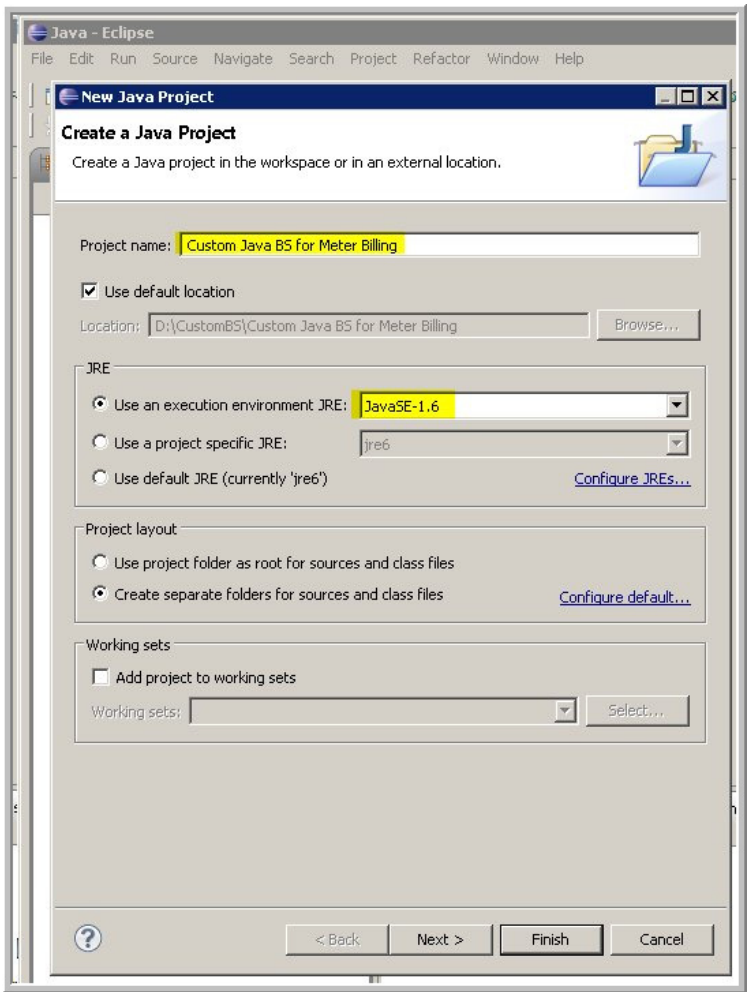
To set up the eclipse environment you must create a java project in a new or existing eclipse workspace.

Pre-requisites

- Eclipse 3.5+ IDE for Java Developers
- JDK 1.6.20
- spl-mcp-BS-[current release].jar (Delivered separately along with MDT Runtimes) For example, spl-mcp-BS-2.2.0.jar

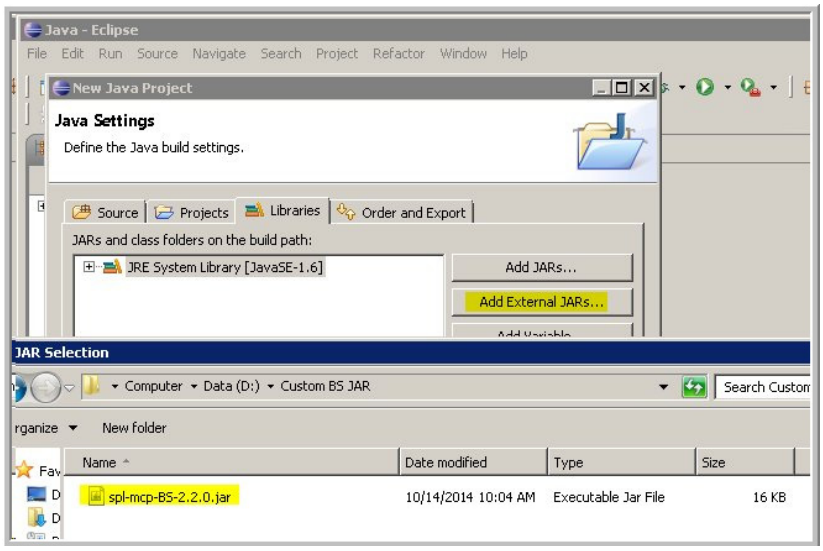
Creating a Java Project (in a new or existing Eclipse workspace)

1. Create a New Java Project using the File -> New Project Wizard.
2. Type in a project name, set the execution environment JRE to JavaSE-1.6 and click **Next**.



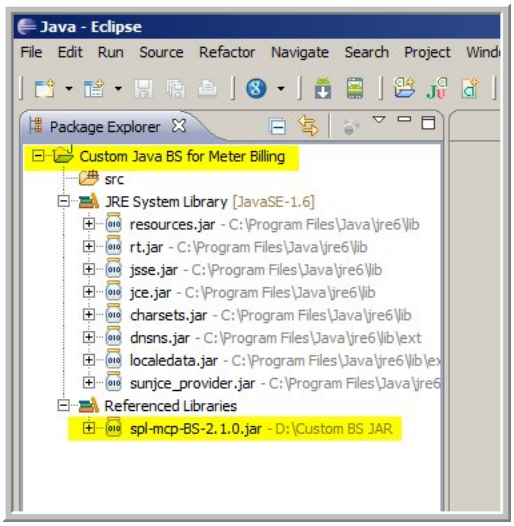
3. Click the **Add External Jars** button and select the **Libraries** tab.

4. Add the **spl-mcp-BS-[current release].jar** as an external JAR
For example spl-mcp-BS-2.2.0.jar.

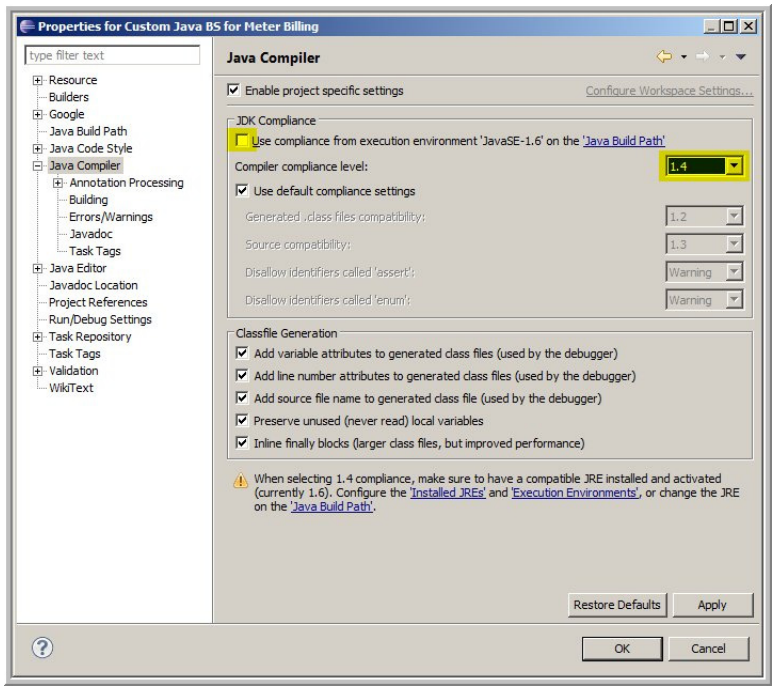


5. After the Import, Click "**Finish**" to create the Java Project.

The Java project is now successfully created with the imported Java BS API JAR file



6. Right Click -> **Project properties** -> **Java Compiler** to change the Java project's code compliance to JDK 1.4 (to ensure same code that's written in this project can work on Windows Mobile/Laptop and Android).



The Java project is successfully setup.

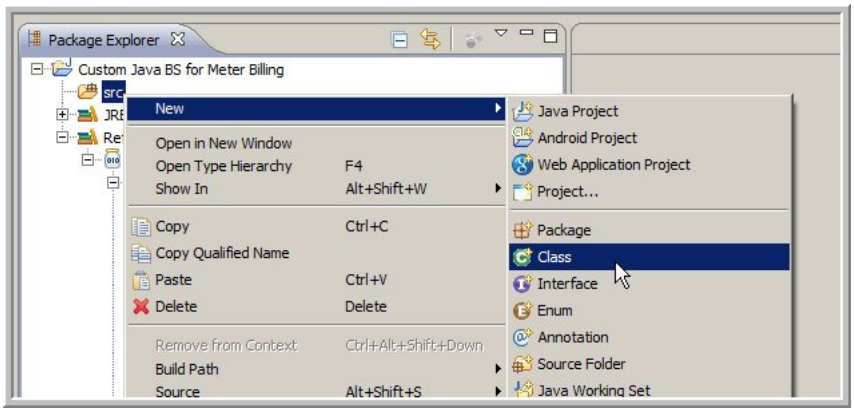
Write the Java Business Services

To write the java business services you must do the following:

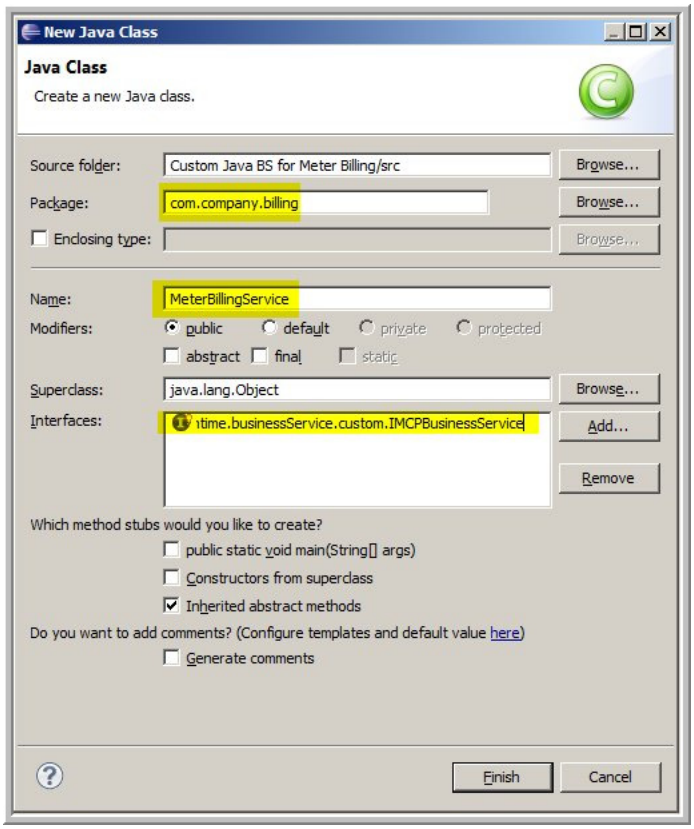
- Code the Business Service Implementation
- Create Business Service Metadata

Coding the Business Service Implementation

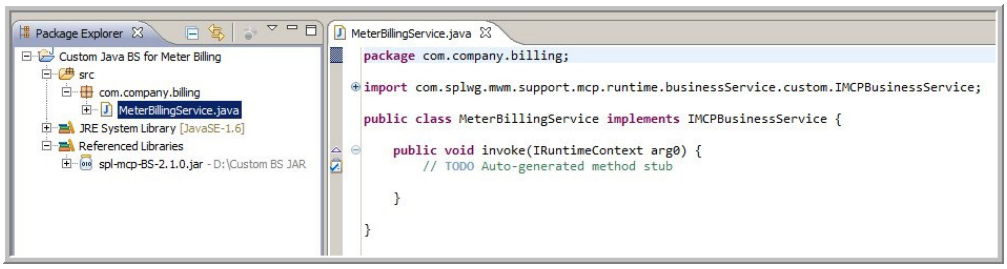
Creating a new Java Class file to write the Java Business Service.



1. Enter the package name, the class name, and **IMCPBusinessService** as an interface this class implements and click "**Finish**".



A Java Business class has been created as below.



2. Implement the Java Business Service using the Custom Java Business Services APIs.

```
package com.company.billing;

import
com.splwg.mwm.support.mcp.runtime.businessService.custom.BusinessServiceError;
…… Other imports here.

public class MeterBillingService implements IMCPBusinessService {

          // This will store the RuntimeContext instance for the runtime.
          // This will hold a reference to the XML DOM
          IRuntimeContext mContext = null;

          // Provides access to MDT JBS API methods.
          IMCPCallback mcpCallback = null;

          // Just hard-coding the value for the sake of the example
          private static int COST_PER_UNIT = 4;
```

```
	private static Class clazz = MeterBillingService.class;

	/**
	 * Mandatory method - is a start point for any Custom Java Business
Service.
	 */
	public void invoke(IRuntimeContext context) {
		mContext = context;
		mcpCallback = context.getMCPCallback();

		mcpCallback.logInfo("Reached MeterBillingService ", clazz);

		double billAmount = calculateBill();

		mcpCallback.logInfo("Bill Amount = " + billAmount, clazz);

		mcpCallback.logInfo("End of MeterBillingService ", clazz);

	}

	// Setter to set the Runtime Context (used only in Unit testing -
where a RuntimeContext)
	public void setRuntimeContext(IRuntimeContext pContext) {
		mContext = pContext;
	}

	public String[] getUserInformation() {
		String[] details = mcpCallback.getUserInfomation();
		mcpCallback.logInfo("User name = " + details[0] + " Password
= " + details[1], clazz);
					return details;
	}

	public double calculateBill() {

		DataElement currentElement = mContext.getJBSDOM();
		DataElement bsData =
currentElement.getChildElement("bsData");
		double meterReadStart =
bsData.getChildElement("meterReadStart").getDoubleValue();
		double meterReadEnd =
bsData.getChildElement("meterReadEnd").getDoubleValue();
		double billAmount = COST_PER_UNIT * (meterReadEnd -
meterReadStart);
		DataElement meterBillAmount =
bsData.getChildElement("meterBillAmount");

		if(meterBillAmount == null) {
			if(mcpCallback.isEligibleForDebug()) {
				mcpCallback.logDebug("Output Element -
meterBillAmount not previously found, adding it now", clazz);
			}

bsData.addChildElement("meterBillAmount").setDoubleValue(billAmount);
		}  else {
			if(mcpCallback.isEligibleForDebug()) {
				mcpCallback.logDebug("Output Element -
meterBillAmount previously created, skipping creation", clazz);
			}
			meterBillAmount.setDoubleValue(billAmount);
		}
```

```
                    mcpCallback.logInfo("Reached MeterBillingService ", clazz);
                    return billAmount;
                }
}
```

Creating Business Service Metadata

1. Open the Online ORS web application.
2. Create a new Business service by navigating to:

   Admin Menu → B → Business Service (+)
3. Enter a business service name and other details.

   Business Service Name/Code – Prefixed by **CM**.

   Example: CM-MeterBillingBusinessService

   Description and Detailed Description – Describe the business service's usage.

   Service Name – For developing Custom Java Business services, always use **M1-MCPCUSTOMBS**.
   This is somewhat different from how the online Oracle Real-Time Scheduler application business services are written in which the business service implementation class and its service program are plugged-in through this field.

   Application Service – You can use **F1-DFLTAPS** (this is not used on the MDT Runtime).
4. Click on the "**Schema**" TAB to provide a Business service schema. Through this schema you will map this business service metadata to the Business service Java Implementation. There are two elements in the schema which are important and should always be included.

   bsData – Any input/output for this business service is passed inside this.
   An XPath example – "ZZ-MeterBillingBusinessService/bsData/meterReadStart"

   bsClassName – This mentions the Java class name (without the .java or the .class extension). The class-name should be mentioned in the **"default"** attribute and is mandatory. In this example you can see that the Java class created in the coding phase is used here. "**com.company.billing.MeterBillingService"** is the Business Service Java Class Name.

```
<schema>
<bsData type="raw" mapXML="RAW_XML'/>
<bsClassName default="com.company.billing.MeterBillingService"
required="true" mapField="JAVA_CLASS_NAME"/>
</schema>
```

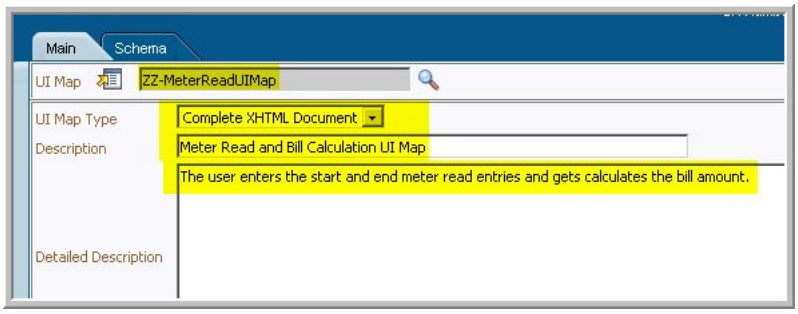
Example Usage of the Business Service

1. Create a new UI Map which uses the business service.
   Admin Menu → U → UI Map (+)

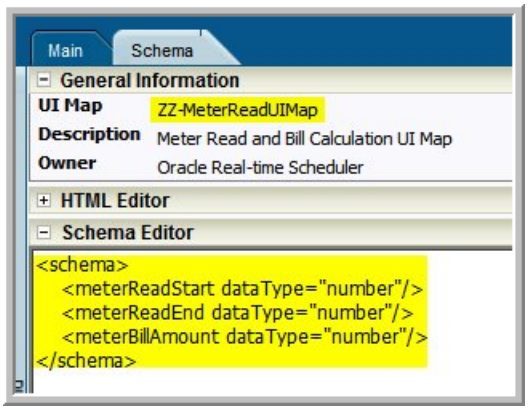
   Enter the UI map details

   - UI Map Name/Code – usually starts with CM – Example: CM-MeterReadUIMap

- Description and Detailed Description – Describes the usage of the UI map.



2. Enter the UI Map schema.



3. Generate HTML for an Input Map using the dashboard zone by clicking the **Input Map** button.

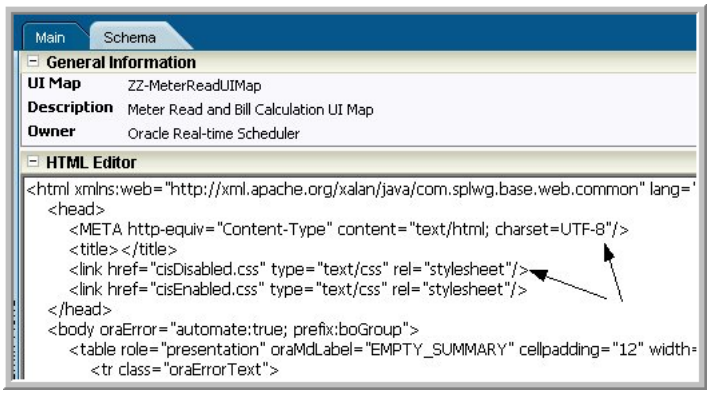
You can now see that the HTML is generated in the "HTML Editor" section. Click "**Save**" again.

The generated HTML has to be modified to make it XHTML complaint by closing all unclosed HTML tags.

Example:

Original<link href="cisDisabled.css" type="text/css" rel="stylesheet">

New<link href="cisDisabled.css" type="text/css" rel="stylesheet"/>

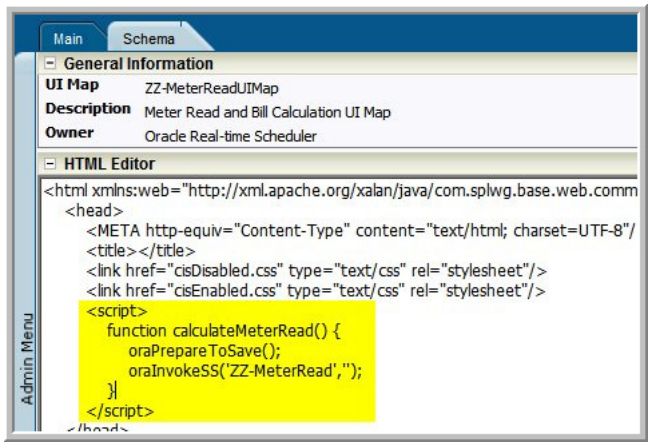


4. Modify the **HTML** buttons to call the Custom Business Services.

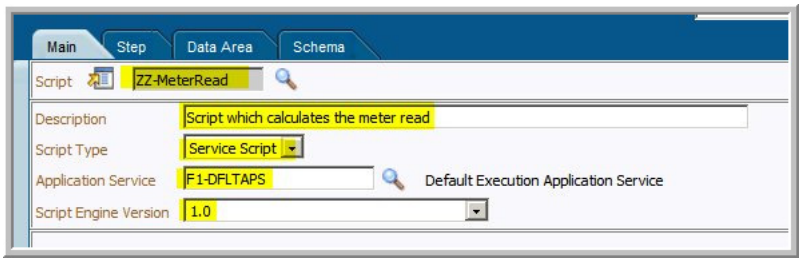
```
<td class="oraSectionStart oraEmbeddedTable" colspan="2">
  <table role="presentation" oraMdLabel="EMPTY_SUMMARY" cellspacing="2">
    <tr>
      <td><input onClick="calculateMeterRead();" value="Calculate Meter Bill Amount" class="oraButton" type="button"/>
    </tr>
  </table>
</td>
```

5. Write Java Script method to calculate Meter Read (which calls a service script).

   Note: oraPrepareToSave() suppresses the "unsaved data changes warning".

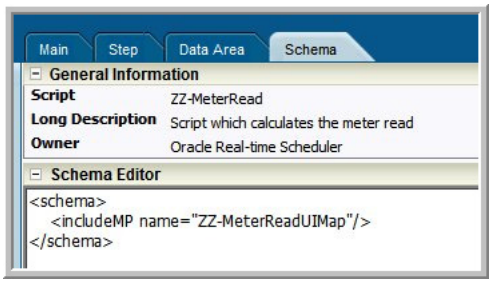


6. Create a service script that the UI map calls for Meter Read calculation.



   Since the service script calls the BS, include the BS schema.

   Service script has the same fields as the UI map, so Include Map is used here. For this reason, it is not necessary to explicitly include the UI Map in the "Data Area" section above.

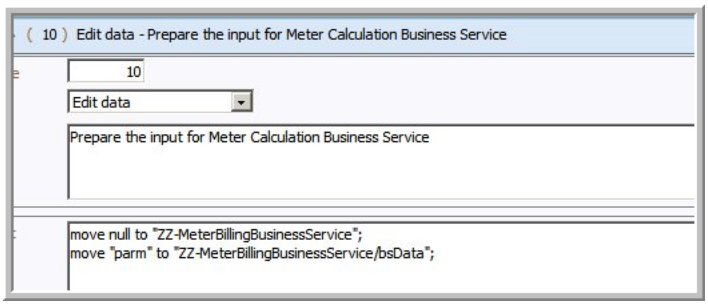


The include helps pass the data between the UI map and the service script and back to the UI Map. At runtime, the service script schema resembles the example below:

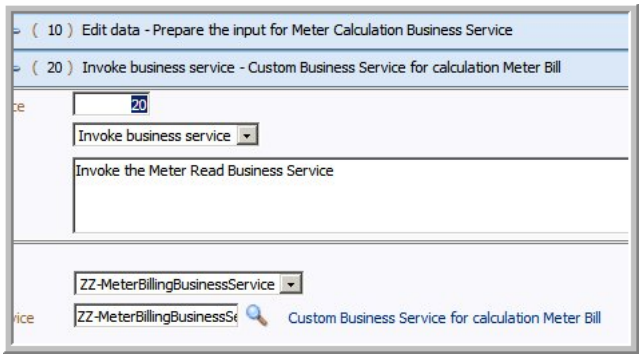
```
<schema>
  <ZZ-MeterBillingBusinessService type="group">
    <bsData type="raw"/>
    <bsClassName required="true" default="com.company.billing.MeterBillingService"/>
  </ZZ-MeterBillingBusinessService>
  <parm type="group">
    <meterReadStart dataType="number"/>
    <meterReadEnd dataType="number"/>
    <meterBillAmount dataType="number"/>
  </parm>
</schema>
```

7. Add script steps to set the input data to the BS and call the BS and fetch the output to be passed to the UI Map.

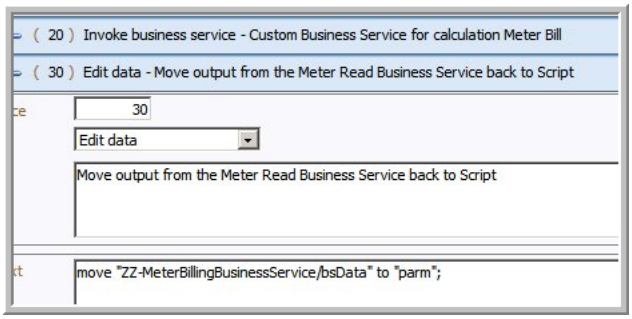
   - Pass the inputs received from the UI Map to the Business service.



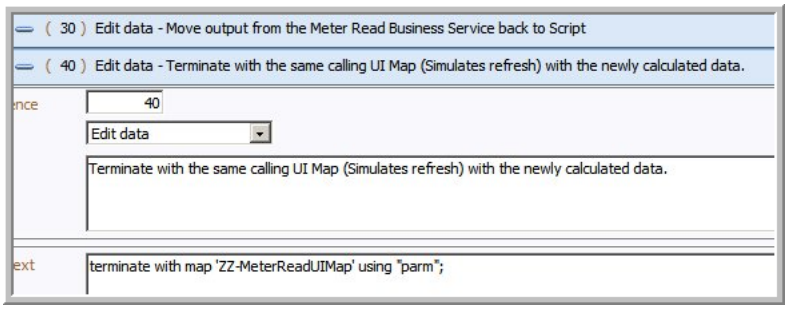
   - Call the custom meter business service.



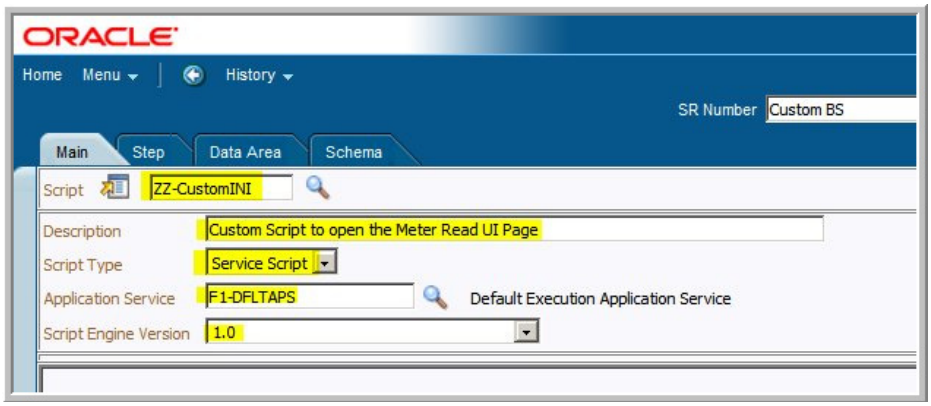
   - Get the BS output and set back to script schema.



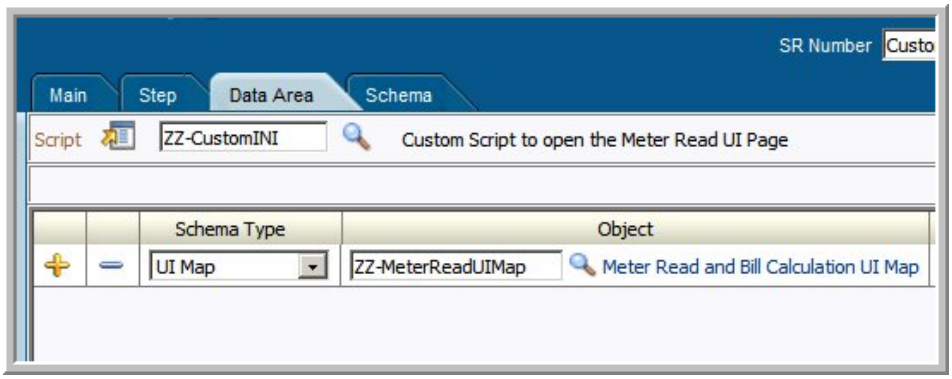
- Terminate with the same UI Map by passing the script schema to the UI map.



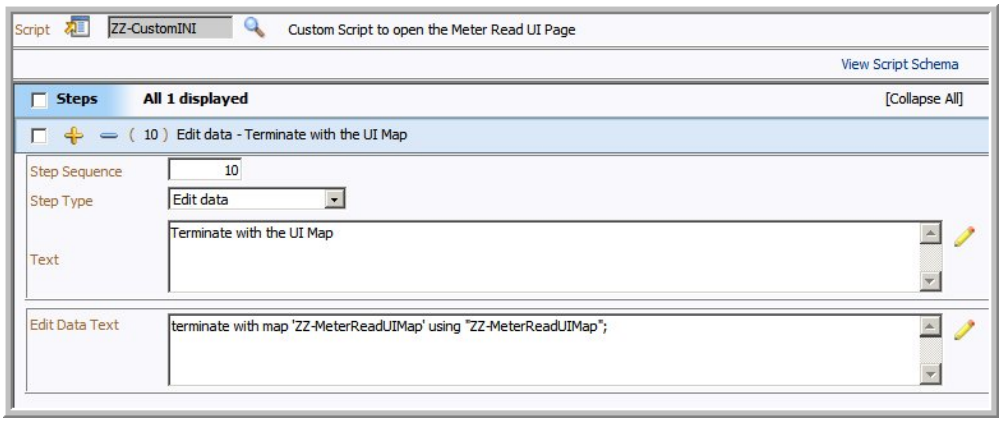
8. Create a new script which is used as the initial service script.



9. Add the UI map created to the script.



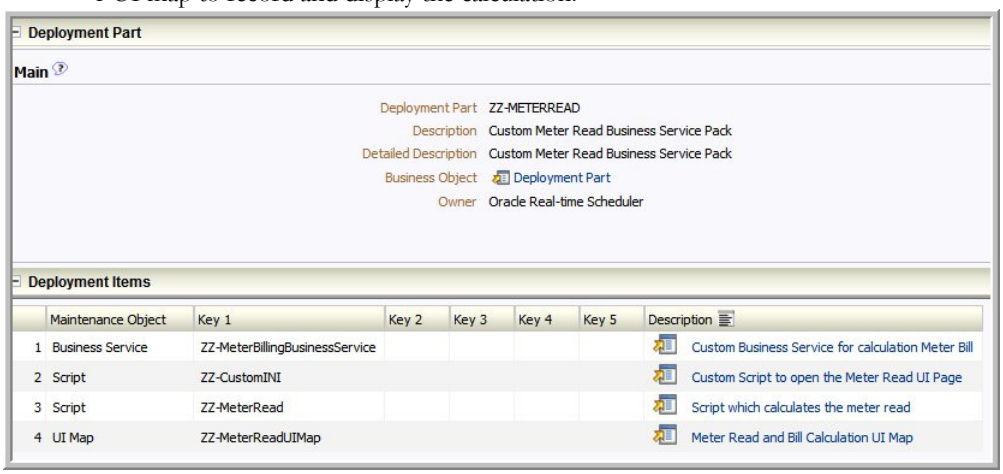
10. Create a script step to terminate with the map.



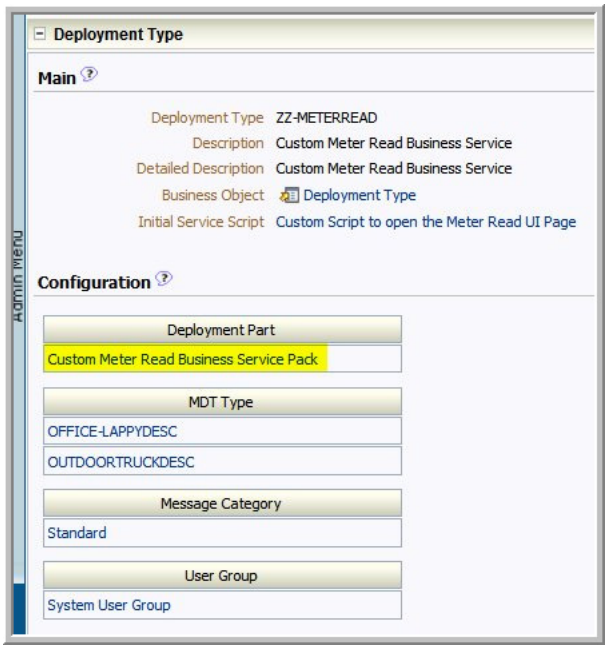
11. Now that all the metadata for the example is created, prepare this for Deployment.

- Create a Deployment Part to include the items created

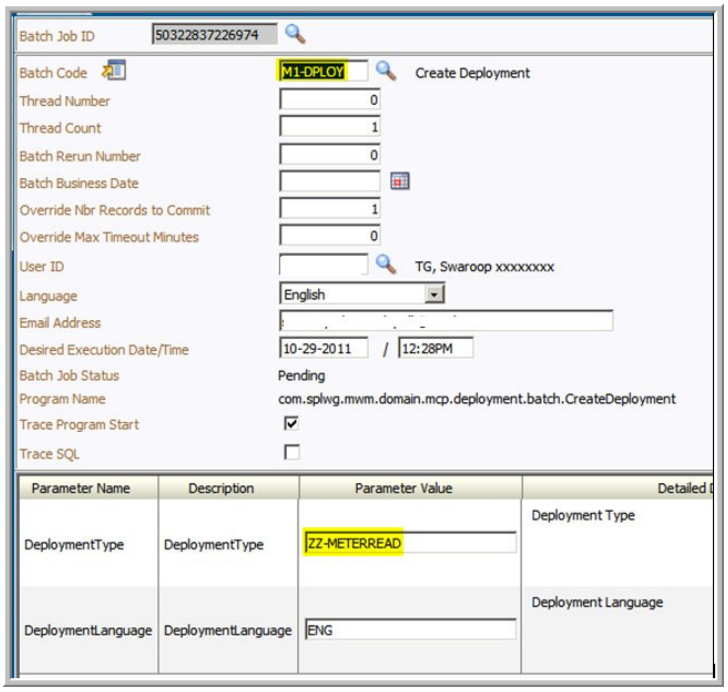
- 2 Service scripts (Initial Service Script & Script to call BS)
- 1 Business service – The custom business service that was created
- 1 UI map to record and display the calculation.



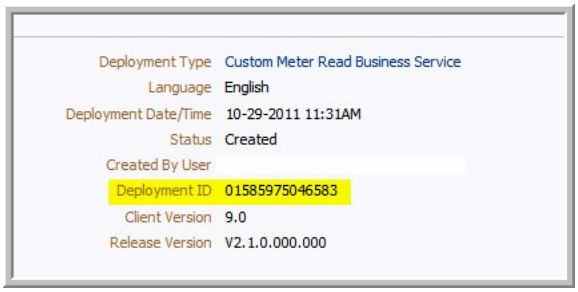
12. Create a Deployment Type with this Deployment Part.



13. Run the Batch Job to generate Deployment for this Deployment Type.



14. Activate the deployment after the batch job status is "ended" and there are no errors observed in the Batch Run Tree.



Compile and Build the Custom Java Business Services

To compile and build the custom Java business services you must build the Java code into JAR and APK.

Building the Java code into JAR and APK

MDT runtimes require the Custom Java business services compiled in different formats.

- Windows Mobile MCP – Supports multiple JAR files
- Windows Laptop MCP – Supports multiple JAR files
- Android MCP - Supports only 1 APK, so classes from multiple JARS have to be merged into a Single APK file.

You can maintain all 3 using ant build scripts in the same project location.

Windows Laptop – Compiling the Business Service Code into a JAR File

This simple Ant script compiles the Java code into a JAR File and copies it into the c:\ORSApp\ext. The significance of the "ext" folder is explained in the Test Inside MDT Runtime section.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project basedir="." default="buildjar" name="MCP-BS-Builder">
<description>
Ant build script to build Custom BS implementation's JAR
</description>
    <property name="output.dir" value="bin" />
    <property name="target.dir" value="C:\MWMORSApp\ext" />
    <property name="jar.name" value="meter-read" />
    <target name="buildjar">
        <delete file="${target.dir}/${jar.name}.jar" />
        <jar destfile="${target.dir}/${jar.name}.jar"
basedir="${output.dir}" />
    </target>
</project>
```

Windows Mobile – Compiling the Business Service Code into a JAR File

This simple Ant script compiles the Java code into a JAR File and copies it into the device's ext folder.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project basedir="." default="buildjar" name="MCP-BS-Builder">

<description>
Ant build script to build Custom BS implementation's JAR
</description>

    <property name="output.dir" value="binCVM" />
    <property name="target.dir" value="extCVM" />
    <property name="jar.name" value="meter-read" />
<property name="cvm.dir" value="D:\Custom BS JAR\CVM" />

    <target name="clean">
        <delete dir="${output.dir}" />
        <mkdir dir="${output.dir}"/>
    </target>

    <target name="compile">
       <javac srcdir="src"
         destdir="binCVM"
         compiler="javac1.4"
         debug="on"
         debuglevel="lines,vars,source"
```

```
            source="1.4"
            target="1.4"
            executable="C:\Program Files\java\jdk1.6.0_25\bin"
            failonerror="true"
            listfiles="true"
            >
          <bootclasspath path="${cvm.dir}\lib\btclasses.zip" />
          <classpath path="D:\Custom BS JAR\spl-mcp-BS-
2.2.0.jar;${cvm.dir}/lib/charsets.jar;$${cvm.dir}/
libcharsets.jar;${cvm.dir}/lib/jaas.jar;${cvm.dir}/lib/
jce.jar;${cvm.dir}/lib/jsse-cdc.jar;${cvm.dir}/lib/
localedata.jar;${cvm.dir}/lib/foundation_jdbc.jar;${cvm.dir}/lib/
sunrsasign.jar" />
          <extdirs path="${cvm.dir}/lib/ext" />

          </javac>
      </target>

      <target name="deploy">
   <exec executable="c:\Program Files\Windows Mobile Developer Power
Toys\/CECopy/cecopy.exe" failifexecutionfails="true"
failonerror="true">
   <arg value="${target.dir}"/>
<arg value="dev:\ORSApp\ext"/>
</exec>
      </target>

      <target name="buildjar">
       <antcall target="clean" />
       <antcall target="compile" />

          <delete file="${target.dir}/${jar.name}.jar" />
          <jar destfile="${target.dir}/${jar.name}.jar"
basedir="${output.dir}" />

          <antcall target="deploy" />
      </target>
</project>
```

Android Runtime – Supports 1 APK file (JAR converted into APK)

This section describes how to compile and test on Android MDT Runtime.

Prerequisites:

- Eclipse Android SDK (the latest version Rev 15)
- Make sure the Android's platform-tools directory is in the system PATH (or you can point the individual executables in the Antd file).

This simple Ant file creates a APK from the class files and then copies it into the Android Emulator. For using this apk with Android device.

Refer to the section titled Test Inside MDT Runtime for more information.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<project basedir="." default="copyToDevice" name="JAR To Dex
Converter">

<description>
Ant build script to build dex files.
</description>
```

```
 <property name="classes.dir" value="bin" />
 <property name="target.dir" value="D:\CustomBS\Custom Java BS for
Meter Billing\extAndroid\" />
 <property name="apk.name" value="meter-read.apk" />

 <target name="buildAPK">
<exec executable="dx.bat">
<arg value="--dex" />
<arg value="--output=${target.dir}/${apk.name}" />
<arg value="--positions=lines" />
<arg path="${classes.dir}" />
</exec>
 </target>

 <target name="copyToDevice" depends="buildAPK">
<exec executable="copyUsingADB.bat">
    <arg value="${apk.name}" />
</exec>
 </target>
</project>
```

Test Inside MDT Runtime

For Laptop and Windows Mobile MDTs, in the above Compiling and Building step we created a JAR file (multiple JAR files are supported too). Android requires an APK file created which was also explained above. (supports 1 apk file only).

If you have used a similar script as shown in the compiling and building step, then you would have noticed that the JAR file (meter-read.jar) is already copied into the MDT Runtime's ext folder.

Setting up Laptop MDT

For a laptop, the default path where the MSI is installed is C:\ORSApp directory. This can be changed based on installation time preference.

In this installation folder you'll find an "**ext**" folder. Copy the meter-read.jar into the "ext" folder.

Custom css should be placed in the ABS_APP_PATH\css folder.

Setting up Windows Mobile

For Windows mobile, the default path where the CAB is installed is \ORSApp on the device.

In this installation folder you'll find an "**ext**" folder. Copy the meter-read.jar into the "ext" folder.

Custom css should be placed in the ABS_APP_PATH\css folder.

Setting up Android

For Android, the folder where the APK has to be copied is different from Laptop and Windows Mobile.

Android Emulator – The provided ant build file copies the APK file into Android's directory – "**/data/data/com.splwg.base.android/ext**". This folder is writable and is accessible in the case of emulator only. If you wish to copy the file manually, you can use these commands.

```
C:\>adb shell
$cd /data/data/com.splwg.base.android/
$mkdir ext
$exit
C:\>adb push meter-read.apk /data/data/com.splwg.base.android/ext/
```

Android Device – The provided ant build script can compile the APK file and it places it in extAndroid folder of the current Eclipse Project. To import the APK file into the folder "/data/data/com.splwg.base.android/ext":

1. Connect the Android device to the computer and go into USB Disk drive mode.
2. Copy the meter-read.APK file into a new the sdcard into a new folder structure ORSApp\ext\.
3. Disconnect the device from the computer (or move into Charge Only mode).
4. Use Oracle ORS Tools shortcut icon to "Import" and the APK is copied across into the required folder /data/data/com.splwg.base.android/ext.

Custom JS and CSS for Android

Java script and CSS files must also be imported into the private Android directory using the tools program provided by base. These custom files must also be provisioned to the device MWMApp/script and MWMApp/css folders. This is not part of a deployment so the files must be pushed with device management software or repackaged in the distribution (APK, MSI, or CAB) or manually copied to the folders.

Once the files are in the proper place, they are always automatically added to the main template container (MDT_TemplateContainer_en.html). Any css should be availableglobally as well as the JS.

To use cm.js and cm.css in Android:

1. Mount the Android's SD Card to the desktop PC.
2. Create a new folder on the SD Card \MWMApp\.
3. Copy the cm.js into the folder -> \MWMApp\scripts\cm.js.
4. Copy the cm.css into the folder -> \MWMApp\css\cm.css.
5. Unmount the Android's SD Card from the desktop PC.
6. Use Android Export/Import Tool and use "Import" Action.
   With these steps complete, CM.JS and CM.css will be available in all the UI maps, but not the initial registration/login/deployment pages.

   Note: The "Copy" steps can be done via a command line (ADB.commands) or via device management software.

To import the MDT tag and URL:

The MDT tag and URL can be imported in two ways:

- Add them to the SDCard at MWMApp\files\MDT.properties and import.
- Use the "MCP upgrade function" which uses the values placed at the SD Card location (mcpbackup\BackupMDT.properties) if the properties are not already typed in or available.

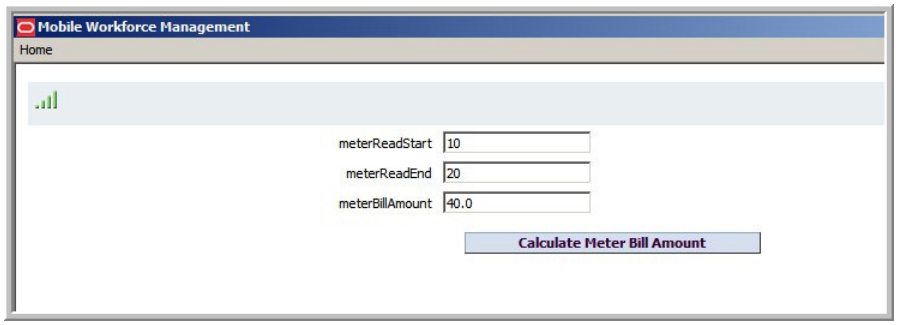
Using MDT Runtime after the Initial Setup

After the initial setup steps as explained above, use the MDT Runtime as follows:

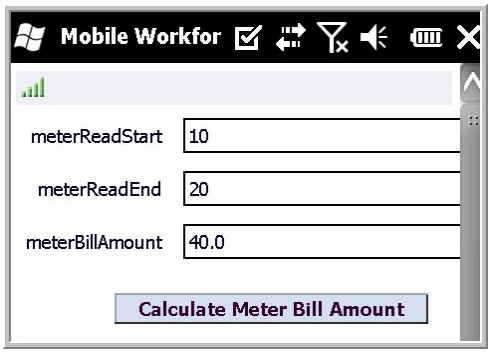
- For all the MDT Runtimes, the application is launched and the new Deployment created in the step Creating Business Service Metadata is downloaded.
- After the download, the usage of this business service is transparent and will work as coded into the UI Maps like any other Business service called in the system.

- Entering "meterReadStart" and "meterReadEnd" and clicking the **Calculate Meter Bill Amount** button will calculate the "meterBillAmount" as 40.0 (20 – 10 * 4).
  This calculation was done using the Meter Read Custom Java Business.
- The test can be repeated multiple times by modifying the start and end values.

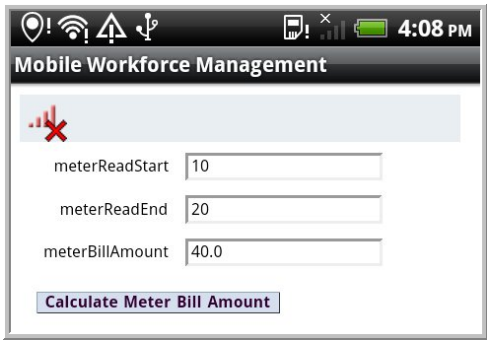
Laptop



Windows Mobile



Android



JUnit Test the Java Business Services

To complete JUnit testing of the Java business services you must:

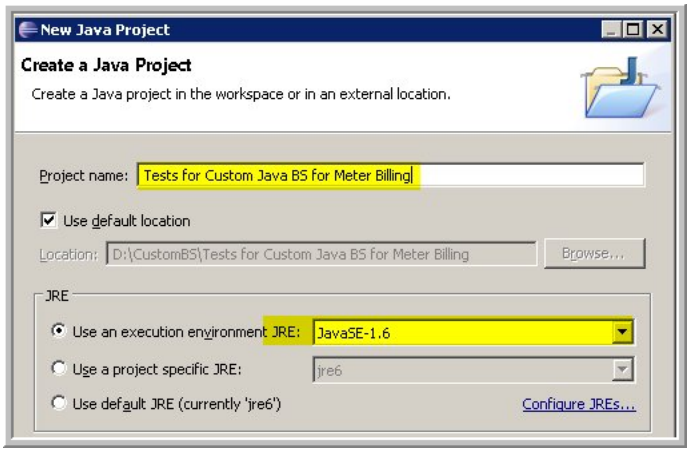
- Create the JUnit Project
- Write a JUnit Test Case

Creating the JUnit Project

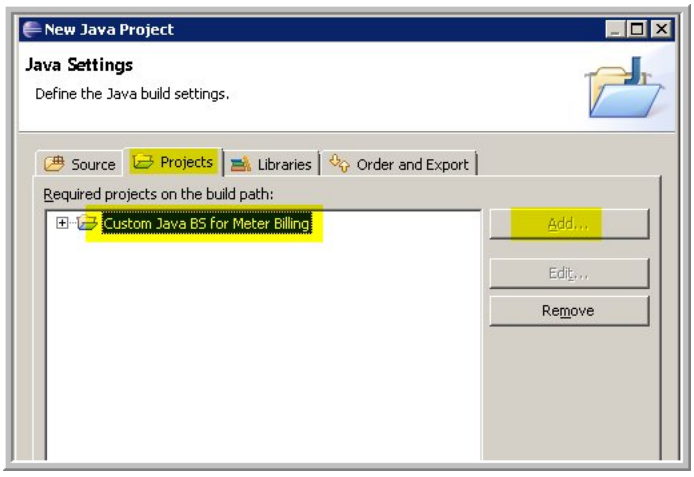
This section provides steps for creating a new Java Project for writing Junit tests.

1. Enter a project name.

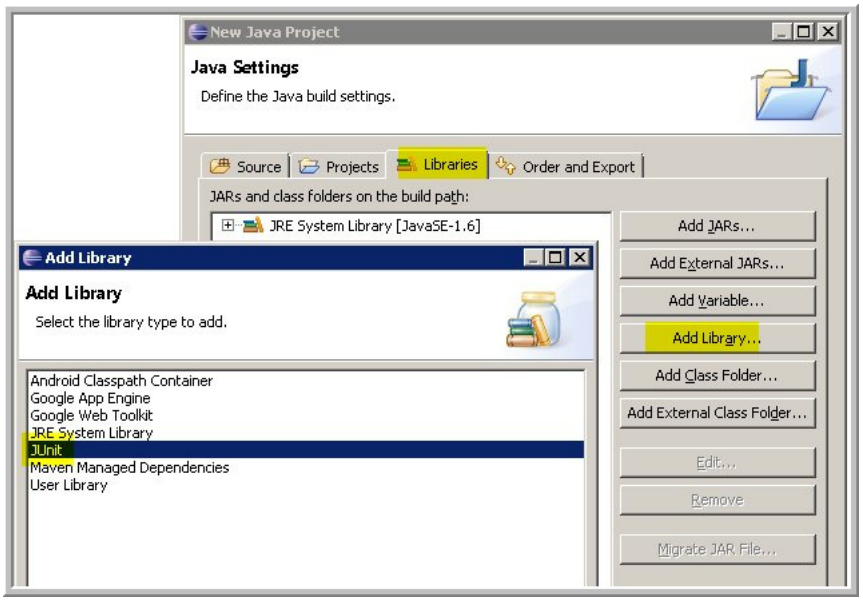
   (For example: Tests for __original project name __)



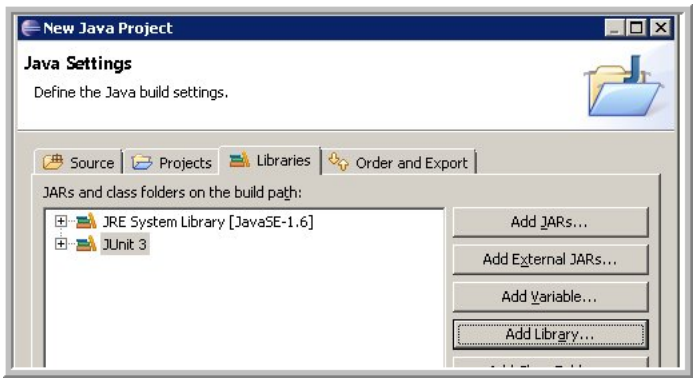
2. Add the original project as a dependency for the new Test project.



3. Add the JUnit3 Library to the current project from the "Libraries" tab.



4. After adding the Junit3 Library, click "**Finish**" and the project is ready.

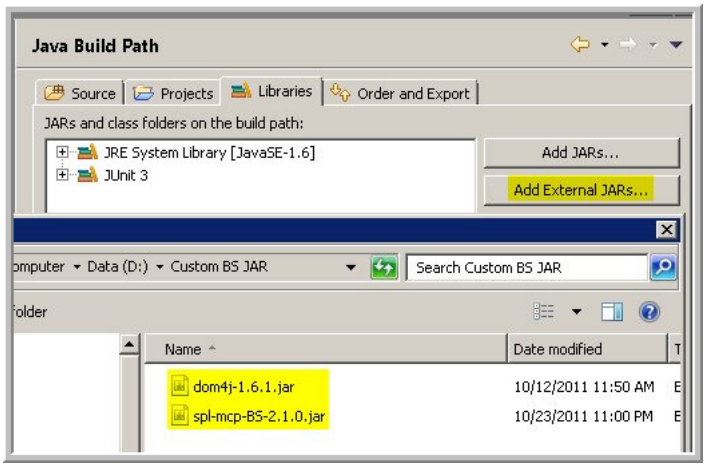


5. Add the dependency JARS to the JUnit project.

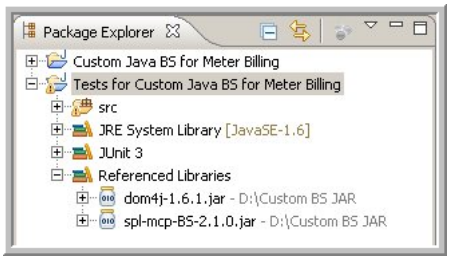
   MDT Java BS API - spl-mcp-BS-[current release].jar – As used to create the Java business service.
   For example, spl-mcp-BS-2.2.0.jar

   DOM4J - dom4j-1.6.1.jar – Can be downloaded from DOM4j's official download page

   http://sourceforge.net/projects/dom4j/files/dom4j/



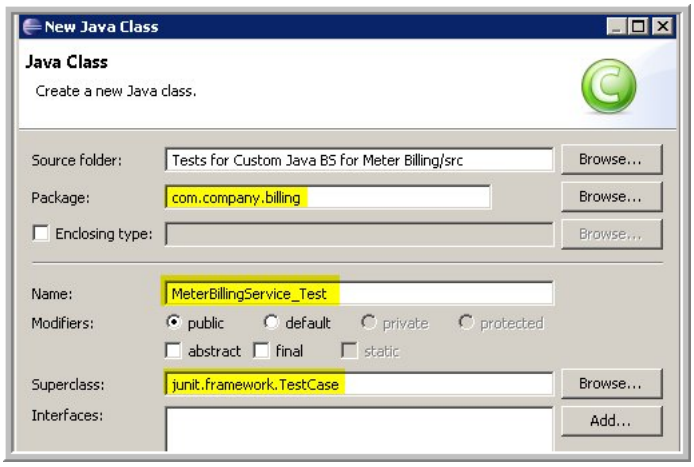
The JUnit test project is now ready to write JUnit test cases.



Writing the JUnit Test Case

1. Create a new Java Class with the same package as the Original Business service class.

   The new Java class is a JUnit java class, so it should extend JUnit's TestCase class.



2. Write a MCPCallbackMock Custom implementation class like below.

   This mock class can over-ride any methods explained in the IMCPCallback section. For example, here the implementation over-rides some of the logger methods to print to System OUT and return sample user credentials to the test case.

```
package com.company.billing;
import
com.splwg.mwm.support.mcp.runtime.businessService.custom.MCPCallbackMo
ck;
public class MeterReadMCPCallbackMock extends MCPCallbackMock {

@Override
public String[] getUserInfomation() {
return new String[] {"SAMPLEUSER", "SAMPLEPASSWORD"};
}

@Override
public boolean isEligibleForDebug() {
return true;
}

@Override
public void logDebug(String message, Class classReference) {
System.out.println(classReference.getName() + " DEBUG: " + message);
}

@Override
```

```
public void logInfo(String message, Class classReference) {
System.out.println(classReference.getName() + " INFO: " + message);
}
}
```

Create the actual test case which calls the Meter Bill amount business service in the JUnit test environment.
The test case uses the new MeterReadMCPCallback mock class created above. If the business service does not call MCP APIs (i.e does not use MCPCallback methods), then one can use the default provided: MCPCallbackMock class instead.
The test case has 2 test methods which test the public methods of the business service. The test case input is a "sample" input as how the business service Java class receives the input in MDT Runtime and then calls the calculateBill() method. The output is read through the Business service's output XML and is validated.
This is how the Java Business Service can be tested outside the MCP Runtime.

```
package com.company.billing;

// Imports go here.

public class MeterBillingService_Test extends TestCase {

public void testMeterBillCalculation() {

MeterBillingService meterBillingService = new MeterBillingService();
RuntimeContextMock context = new RuntimeContextMock();
MeterReadMCPCallbackMock mcpcallback = new MeterReadMCPCallbackMock();
context.setMCPCallback(mcpcallback);
context.setJBSDOM("<root>" +
"<bsData>" +
"<meterReadStart>10</meterReadStart>" +
"<meterReadEnd>20</meterReadEnd>" +
"</bsData>" +
"</root>");
meterBillingService.setRuntimeContext(context);
double billOutput = meterBillingService.calculateBill();
assertTrue("Bill Amount returned by API = " + billOutput, billOutput ==
40);

            DataElement out = context.getJBSDOM();
        DataElement meterBillAmount =
out.getChildElementForXPATH("bsData/meterBillAmount");
double bsOutput = meterBillAmount.getDoubleValue();
assertTrue("Bill Amount returned by BS Output = " + bsOutput, bsOutput
== 40);

}

public void testUserDetails() {

MeterBillingService meterBillingService = new MeterBillingService();
RuntimeContextMock context = new RuntimeContextMock();
MeterReadMCPCallbackMock mcpcallback = new MeterReadMCPCallbackMock();
context.setMCPCallback(mcpcallback);
context.setJBSDOM("<root>" +
"</root>");
meterBillingService.setRuntimeContext(context);

String[] userInfo = meterBillingService.getUserInformation();
assertTrue("Required User found ", "SAMPLEUSER".equals(userInfo[0]));
```

```
assertTrue("Required Password found ",
"SAMPLEPASSWORD".equals(userInfo[1]));

}
}
```

Java Business Service API Reference

This section explains all the Classes and APIs provided as a separate MDT Java Business JAR which helps in the implementation of MDT Custom Business Services. The custom business service implementations shouldn't refer to any other classes in MCP Runtime except for these classes.

The development environment is the same for developing for all the MDT Runtimes i.e WM/ Laptop and Android too. In the case of Android, the custom BS class files will be converted into an APK file.

IMCPBusinessService (Custom BS Classes should implement this)

This interface contains one method with the following signature

void invoke(RuntimeContext context)

The custom business services will provide an implementation for this method by using the various other Custom Business Service APIs explained below to operate on the DOM.

BusinessServiceError (Exception)

This exception will be created and thrown from within the custom BS error handling code. The runtime implementation of this exception uses the existing BSException: (**com.splwg.mwm.support.mcp.runtime.businessService**)

There are ways for the user to show an Error to the User :

- BusinessServiceError (String message)

  Uses string input as message that is shown to the use

- BusinessServiceError (Integer msgCategory, Integer msgNumber, String[] params)

  Uses MDT-FormatMessage BS (internal methods) to form a Language specific message.

IRuntimeContext (RuntimeContext Interface)

This interface abstracts out the actual calls to RuntimeContext. An instance is set before the BS call is made. If the BS is executing in a junit environment, then an instance of custom RuntimeContextMock instance is set which contains the appropriate test DOM (DataElement)

RuntimeContext (Input/Output to the Custom BS)

This is passed as an input to the Custom Business Service:

Member Variables

- **DataElement** (DOM structure) - Passed as an input to the Custom BS
- **MCPCallBack** (Utility API) - Helps to interact with MCP Runtime (ex: executeSS, logging)

Methods

- **public DataElement getJBSDOM()** – Returns the DataElement variable.

- **public MCPCallBack getMCPCallBack()** – Returns the MCPCallBack variable.

RuntimeContextMock (Mock Implementation)

This is the default implementation of the RuntimeContext interface. Customers may extend this class to override the default implementation if necessary. This class is only used in custom BS development and testing. It is not used by the MCP Runtime.

Methods

- **setJBSDOM(DataElement)** – Sets the data element to given Data Element.
- **setJBSDOM(String)** – Parses and sets the data element to the given XML.
- **setMCPCallback(MCPCallback)** – Sets the MCP Callback object.

IMCPCallback (Default MCPCallback interface)

This interface defines the callback methods that the Custom BS can interact with. The implementation is provided in the runtime by MCPCallback or MCPCallbackMock based on the execution time. For methods to be implemented, check the MCPCabllback class below:

- **MCPCallback** – When the BS runs as a part of MCPRuntime
- **MCPCallbackMock** – when the BS runs as a test program, an instance of MCPCallbackMock is set to RuntimeContextMock which is passed to the custom BS.

MCPCallback (Utility methods to call MCP Runtime)

This contains utility methods for the Custom Business service to interact with MCP Runtime to execute Scripts, other BS, BO operations, logging, querying the database etc.

Methods

1. **InvokeSS** – Allows custom BS to invoke a SS. The implementation invokes ServiceScriptDispatcher with the Script name and the input DOM**:**

   Parameters:

   - Script Name - (input) - String
   - Data Area – (input/output) - JBS DOM

   Example:- MCPCallback.invokeSS("M1-MCPTskLst", dataElement);

2. **InvokeBS** – Allows custom BS to invoke a BS. The implementation invokes BusinessServiceDispatcher which translates to executing the appropriate Business service. This can be used to call MCP Business services and another "Custom BS" implementation also.

   Parameters:

   - BS Name - (input) - String
   - Data Area – (input/output) - JBS DOM

   Example:- MCPCallback.invokeBS("M1-MDTGetMDTInfo", dataElement);

3. **InvokeBO** – Allows custom BS to invoke a BO. Based on the action, this will call BusinessObjectProcessor.java's appropriate method add, replace, delete or read. In all the actions, the input DataElement is updated and can be read in the following step**:**

   Parameters:

   - BO Name - (input) - String

- Data Area – (input/output) - JBS DOM
- Action – (input) – String – “READ”, “ADD”, “REPLACE”, “DELETE”

Example:- MCPCallback.invokeBO(“M1-Assignment”, dataElement, “ADD”);

4. **QueryService** – Allows custom BS to query BO XML. This will directly call the MCP Query Business service’s supporting Java methods for faster result (instead of going through the Business Service Dispatcher).

   Parameters:

   - MO ID – (input) - String
   - List of Where Clause Name – (input) - Array [Where Names]
   - List of Where Value Pairs – (input) – Array [Where Values]
   - List of Sort Columns / Order Pairs - (input) – Array [Column, Order]
   - List of BO Name / Raw XML - (output) – Array [BO Name, JBS DOM]

   Example:-

   String[] whereClause = new String[] {“boStatus”, “IN_SERVICE”};

   String[] sortColumn = new String[] {“CRE_DTTM”, “ASC”};

   Object[] response = MCPCallback.queryService(“M1_CREWSHIFT” whereClause, sortColumn);

   Response:

   response[0] (String) – M1_CREWSHIFT

   response[1] (JBS DOM - DataElement) –

   <root>

   <shiftId>11223344</shiftId>

   <boStatus>IN_SERVICE</bostatus>

   </root>

5. **GetUserInfomation** – Allows custom BS to access user / pass for current user.

   These details are fetched from MCP Runtime’s NetworkManager.

   Parameters:

   - UserName – (output) – String
   - Password – (output) – String

   Example:-

   - String[] output = MCPCallback.getUserInformation();
   - System.out.println(“Username = “ + output[0]);
   - System.out.println(“Password = “ + output[1]);

6. **Info – Logs message at info level**
   Integrates with the existing MCP Logger framework.

   Parameters

   - message – (input) – String
   - Class – (input) – Class

   Example:- MCPCallback.logInfo(“Info Message to be logged”, CustomBSProgram.class);

7. **Debug – Logs message at debug level.**
   Integrates with the existing MCP Logger framework

   Parameters

   - message – (input) – String
   - class – (input) – Class

   Example:-MCPCallback.logDebug("Debug Message to be logged", CustomBSProgram.class);

8. **Error** – Logs message at error level. Integrates with the existing MCP Logger framework

   Parameters

   - message – (input) – String
   - class – (input) – Class
   - Example:-
   - MCPCallback.logError("Error Message to be logged", CustomBSProgram.class);

9. **isEligibleForDebug() -** Allows developer to determine if log level is set to debug before doing any debug logging. This is helpful on the device side by reducing the memory foot print by skipping the costlier MCPCallback.logDebug() call which requires String input.

   Returns Boolean

   Example:

```
if(MCPCallback.isEligibleForDebug()) {
MCPCallback.logDebug("Debug Message to be logged",
CustomBSProgram.class);
                    }
```

MCPCallbackMock (Mock implementation)

This is the mock implementation of the MCPCallback interface. Customers may extend this class to override the default implementation if necessary. This class is only used in custom BS development and testing. It is not used by the MCP Runtime.

Methods

This class will implement each of the interface methods as no op (empty stub) methods and the customer can override the methods to return specific output. For example, the customer may route all the log calls to System Console instead of to a log file.

Frequently Asked Questions

Determine Whether the Custom Java BS JAR/APK Loaded

This can be confirmed from the logs by such log statements in MDT Debug Log level

- DEBUG: Custom JAR found = file:C:\ORSApp\ext\meter-read.jar
- DEBUG: Successfully loaded 1 JARS from path C:\ORSApp\ext

Determine Whether the Business Service is Invoked Correctly

By logging the Business service input and exit points in MDT Debug Log level MCPCallback.logDebug(). You can check the logs to check if the BS call is successful

Logging Practices

Minimize logging in INFO log level i.e MCPCallback.logInfo().

DEBUG logs are written only in MDT log level DEBUG. To optimize the over-head of extra strings getting created when the Log level is INFO, an example is provided in the API section of this document in which the MCPCallback.isEligibleForDebug() check is used before logging.

Appendix F

Glossary

Acknowledgement Required
Indicates that the crew is required to manually confirm receipt of activities of this type.

activity time window (ATW)
The time period during which an activity can be scheduled. See also effective time window (ETW), preferred time window (PTW), arrival time window (ATW), and service time window (STW)

activity type
Defines the nature of the task to be performed at a specific location.

alert
A situation in which predefined condition(s) have been met that require user attention or intervention.

algorithm
A set of rules for solving a problem in a finite number of steps.

allocate
A user-initiated action that locks an activity to a particular crew shift. See also assign and unassign.

allocation priority profile
See priority profile

alternate ID
A different way of uniquely identifying an activity, in addition to its system-generated primary key. A short and simple alternate ID is typically used if an activity is dispatched to a crew by voice and the activity's system-generated key is too long to communicate verbally.

application service
Defines the actions (access modes) supported by a securable function in the system. An application service exists for every transaction and zone in the system, and every business object must reference an application service. Application security is defined by granting or denying user groups access to specific application services.

appointment
An activity for which an arrival time window has been set in advance.

Appointment Booking Group
Defines valid time windows for appointment booking.

arrival time window
A period of time within an activity's effective time window (ETW) during which the activity is allowed to be scheduled, based on applicable restrictions or conditions. See also effective time window (ETW), preferred time window (PTW), and service time window (STW)

assign
An action, performed by the scheduler, that links an activity to a particular crew shift. Assigning is done only by the system. A user can unassign an activity, which unlinks it from its current crew shift and makes it available for rescheduling to any other crew shift, and then the user can allocate it to a different shift; however, a user cannot assign activities.

assignment
A crew's copy of an activity. It is used by the crew to record status changes and completion information. If the crew uses a mobile data terminal (MDT) to work their schedule, a copy of the assignment is dispatched to the crew's mobile device.

assignment in progress
An activity status indicating that a copy of the activity, referred to as an assignment, has been created. The activity remains in this status until the crew completes the assignment or returns it.

assumed equipment
Equipment (capabilities) typically associated with vehicles of a particular type.

assumed skills
Skills (capabilities) typically associated with mobile workers of a particular type.

auto dispatch
A system feature that provides automatic dispatching of activities to crews once they are logged on. Also refers to an activity type attribute that enables or disables auto dispatch for specific activity types. The Drip Mode attribute defines auto dispatch settings at the crew shift level.

base service areas
Service areas within which a mobile worker typically works.

base time
The time zone defined on installation options, which is typically the server time zone.

being scheduled
An activity status indicating that the activity has been sent to the scheduler and is in the process of being scheduled. An activity also transitions back to this status if its current assignment to a crew has been recalled or returned.

BPA
Business Process Assistant. See BPA script.

BPA script
A set of commands that run on the client's browser and guide a user through a business process.

break
A planned time period within a crew shift during which the crew is eligible for taking a break. During a break, the crew may not carry out work or travel.

break duration
Duration of the break.

break window duration
Duration of the period of time during which the actual break can be taken. For example, a 15-minute morning break might have a 60 minute window during which the break can be taken.

calculated
Computed; arrived at or determined by mathematical calculation.

cancel
An action that typically terminates the lifecycle of a business entity. For example, canceling an activity stops the effort of scheduling and performing it.

cancelled (status)
The status resulting from a cancel action. This is typically a final state. For activities, a cancelled status indicates that the host system or the dispatcher has cancelled the activity and it is no longer being considered for scheduling. For crew shifts, a cancelled status indicates that the shift was manually cancelled.

capability
An ability associated with a resource that qualifies the resource for a certain type of work.

capability type
A type of ability that qualifies a resource for a certain type of work. Capability types may define human skills or equipment.

class
A broad grouping or categorization.

clustered tasks
If a crew shift contains multiple tasks of the same class and status that are back-to-back (with only a short period of idle between), then the system collapses the individual tasks into one clustered task in the **Scheduling Gantt** task area. You cannot take action on clustered tasks; you must first expand the shift to uncluster the tasks.

collision
A clash; a conflict; a meeting of objects that causes an adverse impact. See collision detected.

collision detected
An activity status that occurs when more than one completed assignment is received by the server for the same activity.

Common Dispatching Functionality
A component of Oracle Real-Time Scheduler that provides functionality for dispatchers to effectively monitor and manage workforces and activities.

Common Dispatching Interface
The primary point of user interaction for the **Common Dispatching Functionality** (CDF).

complete
An action that finalizes the processing of a business entity and ends its lifecycle.

complete (activity status)
An activity status indicating that the requested work has been done and completion details are available to be sent back to the requesting host system. A completed assignment has been received (one and only one) for an activity and all other assignments for that activity are finalized. The activity remains on the mobile device until logoff. If an activity was in a **Collision Detected** status

and the dispatcher has identified which completed assignment to use, the activity transitions to the **Completed** status.

complete (shift status)
A crew shift status indicating that the crew is done working. This is also known as the End Of Shift. A shift cannot be completed if any of its tasks are not completed or returned.

completion remarks
Comments typically entered by a crew when an activity is completed.

complex activity
A long duration activity that requires more than one crew shift to complete.

contractor capacity
A capacity of work of a certain type, in a certain area and over a period of time that an external contractor agrees to perform.

cost profile
A type of scheduler configuration that establishes a pre-defined set of cost controls. Different cost profiles can be defined to handle different situations and users can actively switch between cost profiles, without shutting down the scheduler, to cause the scheduler to respond based on the defined circumstances. A cost profile is defined as a type of scheduler configuration and then is referenced on the general scheduler configurations that are then applied to schedulers.

crew
A uniquely named group of resources (mobile workers and vehicles) scheduled to perform work.

crew allocation
The specific mobile worker(s) and vehicle(s) assigned to a crew on a particular shift.

crew shift
A planned period of time in which a crew is available to work.

crew shift template
A tool for quickly generating crew shifts based on pre-defined parameters.

crew size
The minimum number of mobile workers needed to perform activities of a particular type.

dashboard
A portal that always appears on the desktop. Its zones contain tools and data used to complete common tasks.

deadreckoning
A method of estimating the current position of a moving vehicle where GPS is not available or updates are received infrequently.

deployment
A specific cut or build of a deployment type for a specific language; a packaged application to be delivered to a mobile device terminal (MDT); also refers to the gathering up of application components (maintenance objects, business objects, UI maps, SS, etc.) and their preparation for delivery to and usage on the MDT.

deployment part
A collection of deployment items, such as maintenance objects, business objects, UI maps, etc.

deployment type
Defines attributes of an application to be deployed, including authorized user groups, supported MDT types, messages used, and objects included.

detected
Discovered; found.

disable
To make unavailable or inactive.

dispatch
An action, either system- or user-initiated, that locks an activity's sequence in the shift schedule and marks it as ready to be sent to the crew. See dispatched status.

dispatched (status)
An assignment status indicating that the assignment has been synchronized to the crew and the activity is now locked to its assigned crew shift and sequence. If required, manual acknowledgement has been given.

dispatch area
A predefined set of service areas and service classes monitored by one or more dispatchers.

dispatcher
An individual responsible for monitoring and managing day-to-day field operations.

drip mode
A crew shift attribute that indicates how the system should handle dispatching of activities to that crew shift. Valid options are:

- **None:** Automatic dispatching is disabled. All scheduled activities must be manually dispatched to the crew's shift by the dispatcher or by another system-invoked process.
- **Auto All:** All scheduled activities are automatically dispatched to a shift once it has started. This applies to any additional activities being scheduled throughout the duration of that shift. This excludes any activities where the Auto Dispatch attribute is set to No for the Activity Type.
- **Standard (Drip Mode):** Only a fixed number of scheduled activities are dispatched to the crew at any given point of time. The fixed number is specified as the Drip Horizon.

drip horizon
The number of activities to dispatch when drip mode is set to Standard or Hybrid.

effective date
The date on which a characteristic, parameter value, or other element becomes effective. See also arrival time window (ATW), preferred time window (PTW), and service time window (STW)

effective time window (ETW)
The time period within which an activity should be scheduled, based on values sent from the host.

emergency task
A task of highest priority, indicating that its scheduling takes precedence over all non-emergency tasks.

en route
An **assignment** or **POU** status indicating that the crew is on its way to the service address or meeting address.

enable
To make available or active.

end odometer
The final odometer reading (total distance traveled by the vehicle) recorded at shift logoff.

equipment
Device or equipment used to perform certain types of work. In Oracle Real-Time Scheduler, equipment is a capability that can be associated with one or more vehicles (resources).

estimated duration
The approximate length of time it should take to complete an activity.

field referenced activity
An activity created by a crew when connectivity problems prevent the crew from receiving an activity and require the dispatcher to voice dispatch it. The FRA references the original activity and, when connectivity is re-established, the FRA is synchronized with that activity on the server.

finalized
Indicates that an object's current status represents the end of its defined lifecycle. For example, in the base system an activity is finalized if it is in a canceled or completed state.

foreign key
A referential constraint between two tables. The foreign key identifies a column or a set of columns in one (referencing) table that refers to a column or set of columns in another (referenced) table.

foreign key reference
Defines the program used to construct a maintenance object's info string. It may also define the transaction used to display, add, and update instances of the maintenance object.

Gantt
A tool that allows dispatchers to view shifts in a timeline and perform actions on activities for those shifts.

Geocode
The longitude and latitude associated with an address, which is used for optimizing routes and displaying crews and activities on a map.

Global Positioning System
A navigation system based on a network of satellites that send out radio signals to pinpoint an entity's location.

hierarchy
Hierarchies group resources, crews and service areas into a hierarchical structure. Field resources can be grouped by the type of work they do, how they are managed, internal versus external, etc. Service areas can be subdivided into territories, regions, offices, etc. You can define hierarchies of up to three levels to establish these categorizations for crews and service areas.

holding scheduler
A holding scheduler is not associated with a schedule manager and therefore does not schedule shifts or tasks. It is designed to capture tasks and shifts that cannot be assigned to any online scheduler. One and only one holding scheduler should exist. Any task or shift attached to a holding scheduler is evidence of a scheduling error.

host external system
A computer system external to Oracle Real-Time Scheduler that sends work orders (activities) to and receives updated information from Oracle Real-Time Scheduler.

idle time cost
A scheduling factor that indicates whether or not crew idle time cost should be considered by the scheduler.

in service
A crew shift status indicating that the crew is considered to be performing work. The crew may be traveling to an activity location, working an activity, or idle between activities. The crew status transitions to In Service automatically when an activity is started or a POU or break is ended. The user can manually transition to this status.

issue
A situation, problem, or potential problem requiring attention or resolution.

issues detected
An activity status indicating that the activity is not valid and processing cannot proceed. Manual dispatcher intervention is required to resolve the issue. Once the problem has been fixed, the status of the activity can transition to Being Scheduled.

key performance indicators
Quantifiable measurements that support dispatcher decision making and in-day exception handling. A KPI may apply to activities and crew shifts that dispatchers are responsible for in their current shift.

late cost
A scheduling factor that defines a cost associated with late arrival to an activity. It is used to discourage scheduling of the activity outside its time window.

lead time
Number of minutes before a shift is planned to start that the reserved capacity is released.

limited time
Indicates an entity or capability that has an effective and expiration date.

location
A predefined physical location, such as a service center.

logoff delay
Number of minutes before the end of the shift that the scheduler should stop scheduling work. This is used to reserve time at the end of the shift for any end-of-shift work.

logon delay
Number of minutes after start of the shift that the scheduler should begin scheduling tasks to the shift. This is used to reserve some time at the beginning of the shift for performing review and preparation work.

location based services
Crew location and tracking services that utilize GPS data obtained from a mobile resource.

maintenance object
A group of tables maintained together within the system.

manual
Occurring as a result of a user action, rather than automatically by the system.

manual acknowledgement
See acknowledgement required.

MapViewer
A tool, based on Oracle Application Server MapViewer, that provides a visual representation of monitored activities, crews, and routes on a map in the Common Dispatch Interface.

message category
A set of application messages.

metadata
Data about data. Metadata may define formatting, validation, possible values, data type, etc.

missing equipment
Equipment typically associated with the vehicle type but missing from this vehicle.

missing skills
Skills that a mobile worker of the specific job type is assumed to have, but this mobile worker does not have.

mobile data terminal
A computerized device typically used to communicate with a central dispatch office.

mobile worker
Refers to any individual, crew member, field resource, field technician, contract resource or any other person or resource completing activities, tasks, or other work.

odometer
An instrument that indicates distance traveled by a vehicle.

on hold
An activity status indicating that the activity was placed on hold, either by the dispatcher or by the system based on a business rule. An activity may be placed on hold because something needs to be done before the order can be worked (e.g., parts, permit, etc.). An activity cannot be put on hold once an assignment has been created (unless the assignments are returned). The activity will remain in this status until a user removes the hold manually or a condition occurs that prompts the system to remove the hold, at which time it will return to its previous status.

on site
An assignment and POU status indicating that the crew is at the service site or event location site.

optimization area/set
An optimization area is a geographic sub-region within a scheduler area. An optimization set is the group of sub-regions in one scheduler area. When the scheduling load for a service area is more than a single scheduler can manage, such as too many tasks, optimization may be further limited to

these smaller subsets of that service area. Optimization sets are configured on the scheduler area using longitude and latitude coordinates to define the boundaries.

out of service
A crew shift status indicating that the crew is not performing work. The crew may be attending a meeting, taking a break or having a vehicle breakdown, etc. When a crew goes out of service, they provide a reason and an estimated duration. At logon, a shift is Out of Service until a task is started and then it switches to In Service. Starting a POU or break automatically sets the crew status to Out of Service. The user may manually transition to this status.

overridden POU
A POU that has been modified manually and no longer matches the POU template from which it was generated.

overridden shift
A crew shift that has been modified manually and no longer matches the template from which it was generated.

overtime cost
A scheduling factor that defines the cost of exceeding the planned shift duration.

pending (shift status)
A shift's initial status when it is first created, before it is considered planned and ready to be started.

pending dispatch
An assignment status indicating that the assignment has not yet been synchronized to the crew's mobile device or has not been manually acknowledged by the crew, if the activity requires manual acknowledgement.

period of unavailability
A period of time in which a crew is planned not to perform work, such as while attending a meeting.

planning horizon
The number of days into the future to plan for crew shifts and periods of unavailability. For example, if the planning horizon is set to 60, the system generates crew shifts and periods of unavailability (POU) sixty days into the future. Generation occurs in a sliding window in that each day generates data for the last day incrementally. At any given time, the number of days of planned shifts and periods of unavailability is equal to this value. Planning horizon should be equal to or greater than the scheduling horizon.

portal
A page in an application that contains one or more zones.

posting action
A posting action is associated with every scheduler registry entry. It identifies the type of change made to an entity that impacts scheduling, and is used to communicate this information to the scheduler manager.

postpone
A user-initiated action that occurs when a crew has not yet started work on an activity but must postpone the scheduled work.

postponed
An activity status resulting from the postpone action.

preferred time window
The time period within which it is ideal to schedule an activity, based on values sent from the host. See also effective time window (ETW), arrival time window (ATW), and service time window (STW)

preview calendar
Calendar based on the changes that have been entered but not yet applied.

primary key
A single column or set of columns in a table that uniquely identifies each row in the table.

priority profile
A predefined set of allocation factors used by the scheduler to prioritize the scheduling of an activity relative to other activities.

queue
A list of items, such as activities, that are awaiting processing.

queued for dispatch
An activity status indicating that the activity has been successfully scheduled to a particular crew shift and is no longer eligible for rescheduling during optimization, but has not yet been synchronized to the crew's mobile device.

recall
An action, typically initiated by a dispatcher, that requests the return of an assignment from its assigned crew.

recommended allocation
One or more mobile workers and/or vehicles identified as appropriate or desirable for a crew shift.

recycle (alert)
An action that reassigns an alert to another dispatcher if the currently assigned dispatcher does not respond to it within a configurable amount of time.

reference time
Time used by the scheduler as input to cost-calculations that are sensitive to the current time. Applies only to time-dependent costs.

relative efficiency
Scheduling factor that defines a mobile worker's efficiency as a percentage of the expected efficiency for mobile workers of this type.

relative speed
Scheduling factor that defines a vehicle's speed as a percentage of the expected speed for vehicles of this type.

reserve capacity
A scheduling concept allowing shifts to reserve a portion of their time for a specific service class of work.

reserve capacity percentage
Percentage of shift time to be reserved for work associated with the reserve capacity service class.

reserve capacity type
Service class for which a percentage of the crew shift's capacity is reserved.

resource
A workforce resource, such as a crew, vehicle, mobile worker, or dispatcher.

resource class
A broad classification of resource. Resource classes include mobile worker, vehicle, crew, and dispatcher, and may include additional user-defined classes.

resource planner
An individual responsible for allocating workforce resources to perform work across multiple shifts.

Resource Planning and Scheduling
A component of Oracle Real-Time Scheduler system that handles resource planning, resource management, service management, and scheduling.

returned
An assignment status indicating that the crew has returned the assignment or the assignment was recalled. This is the end of the assignment's lifecycle; the activity will be rescheduled.

scheduler
A component of the Oracle Real-Time Scheduler system that assigns tasks to crew shifts in such a way as to optimize the use of resources and reduce overall costs. An implementation typically runs multiples schedulers in parallel for performance and backup.

scheduler configuration
Defines parameters used to control the scheduler.

scheduler registry
Contains a record for each object currently owned by a scheduler.

scheduler area
A predefined set of service areas that are designated to be scheduled by a particular scheduler or a number of cooperating schedulers. Scheduler areas can be further segmented by optimization sets. These sub-regions in the scheduler area are used when the scheduler area has too many tasks for a single scheduler to manage.

scheduling horizon
The number of days into the future the scheduler will consider when scheduling crew shifts and activities. Only shifts that have already started or are planned to start within that time frame, as well as activities that may be scheduled during that time, are considered. For example, if the scheduling horizon is set to 21, the scheduler considers shifts and activities for the next 21 days. Oracle recommends that the Scheduling Horizon be kept as short as is practical while still meeting customer business requirements. The optimum value for scheduling horizon strikes a balance between long term planning and short term flexibility.

service area
A logical boundary of an organization's territory that may or may not be based on geography.

service class
A broad categorization of activity types, such as Meter Work or Emergency Work.

service level agreement
A part of a service contract where the level of service is formally defined.

service time window (STW)
A period of time within an activity's arrival time window (ATW) during which it is preferable to schedule the activity. See also effective time window (ETW), preferred time window (PTW), and arrival time window (ATW).

shift
Short version of crew shift.

shift cost
A scheduling factor that defines the cost of utilizing a shift.

shift cost profile
A preconfigured set of shift cost-based factors that affect scheduling decisions.

shift promotion cost
A scheduling factor that specifies the multiplier by which the shift promotion cost for an activity type is increased or decreased. This is used to discourage creation of additional shifts.

shift weekly template
A tool used to define a cyclical weekly pattern made up of crew shift templates.

site address
The address of the actual entrance to the site of service, which may be different than the service address.

site delay
The delay involved in arriving at this site.

Short Messaging Service
A service that allows the interchange of short text messages between mobile telephone devices.

shuffler
A shuffler changes the current solution (schedule). The scheduler operates by repeatedly calling a shuffler to change the current solution and calculating the new solution cost to determine whether the change was beneficial or not. The scheduler employs many shufflers, each with its own specialization.

skill
A capability that indicates the ability to perform certain types of work. Mobile workers have skills; mobile worker types have assumed skills; activity types require skills.

SLA
service level agreement

SLA Flexibility
A scheduling factor that defines the period of time within which an activity must be scheduled to avoid incurring the Preferred Time Factor cost.

Preferred Time Factor
A scheduling factor that defines the fixed cost that will be applied for working an activity outside its service time window's SLA Flexibility period.

snooze (alert)
To stop (sleep) for a certain amount of time before reissuing the alert.

solution
The scheduler (activity-to-shift assignments) generated by the scheduler.

solution cost
The value calculated by the objective function for a schedule. This value is indicative of how well the schedule attains its cost goals. The lower the value, the better.

sort sequence
The order in which items appear in a list.

start odometer
The initial odometer reading (total distance traveled by the vehicle) recorded at shift logon.

status reason
The reason why an entity transitioned to its current state.

suspend
A user-initiated action that occurs when a crew has started work on an activity but cannot complete the work due to time, material or other resource constraints. When the crew suspends an order, they must provide an estimate of the time required to complete the work.

suspended
An assignment status resulting from the suspend action. The assignment's life cycle is not terminated and remains on a crew shift until the assignment is completed, a dispatcher re-assigns the activity to another shift (a new assignment is created), or the crew logs off the shift and the assignment is returned. When a crew is ready to resume work on a suspended activity, it transitions to **En Route** status.

synchronize
To communicate data between the server and the MDT so that both have the same up-to-date information.

task
Any activity, POU, or break that occupies time on a crew shift's schedule.

task class
A classification of task. Task classes include activity, POU, and break. Within each task class, there can be any number of task types.

task type
Defines the attributes of a particular type of activity task, POU task, or break. Examples of activity tasks are Meter Read or Equipment Installation.

template
A pre-defined model or example used as a guide to make other objects. POU templates are used to generate actual POUs. Crew shift templates are used to generate crew shifts.

travel time cost
A scheduling factor that defines a cost associated with crew travel time.

UI map
An entity that contains predefined HTML for building a map zone. Its schema defines the fields whose values are inserted into the map at run time.

unassign
A user-initiated action that unlinks an activity from the crew shift to which it has been assigned by the scheduler.

user-defined
Configured or set by a user rather than being assigned by the system.

vehicle
A non-human mobile resource used by a crew.

window cost
A scheduling factor that defines the preference of an activity's time window relative to other time windows.

work calendar
An object that defines work holidays within a defined period of time.

work profile
Work profiles establish a date and time restriction that can be associated to an activity type. With a work profile properly defined for the activity type, the system ensures that the activity is not scheduled outside of the established time window.

work done
An assignment status indicating that the crew completed the work but has not entered the completion details yet. The crew may enter the details later in the day. This is typical to crews using SMS to work their schedule. This is the end of the assignment's lifecycle. The activity remains in the **Assignment in Progress** status while waiting for the completion details to be entered.

work sequence
The task's sequence within the crew shift.

zone
A section of a portal. Examples of zones include **Query** zones, **List** zones, and **Actions** zones.

Oracle Real-Time Scheduler

Release Notes

October 2013

Release Notes, 2.2.0

Addendum A

Scheduler Configuration

The scheduling process automatically assigns tasks to crew shifts, matching field work requirements to crew capabilities and optimizing the routes traveled by crews. The goal of the scheduling process is to produce the most cost effective schedule or the most efficient schedule.

This section describes how to configure the scheduling component and includes the following topics:

Scheduling Basics

In general, the scheduler controls the following operational areas:

- Shift Operations - how shifts are allocated to tasks.
- Break Operations - when breaks, non-productive tasks, and periods of unavailability should occur.
- Activities Operations - how activities and appointments are handled and by which shifts.

The following sections provide a high level understanding on how the scheduler works and what components requires configuration.

The Scheduling Process

The scheduling process has the following basic inputs:

- **Tasks:** Tasks can be field activities, breaks, non productive administrative tasks, and crew related periods of unavailability (POUs), all of which are incorporated into the schedule.

  If using depot functionality, the scheduler inserts visits to the depot as tasks.

- **Crew Shifts:** Shift data includes crew capabilities, covered service areas and service classes, logon and logoff locations, and shift-based cost factors.

- **Contractor Capacities**: Capacity data includes covered service areas and service classes, as well as capacity amount which may be defined in number of activities or number of hours. This is only applicable if you are using capacity contractors. Refer to Contractor Management Basics for more information.
- **Depots**: If you are using depot functionality, the scheduler can manage shifts which include the location where the crew picks up and drops off goods.
- **Map Data:** All locations within the service territory are represented in the map data.
- **Static Control Data:** This includes cost controls and appointment booking groups.

  Note: A user with the required authority can switch between cost profiles so that the scheduler responds and optimizes work accordingly without shutting down the scheduler.

The scheduling process assigns each task a scheduled date/time on a particular shift of a crew or a contractor capacity if using the contractors functionality. As part of the optimization process, a task may be assigned, unassigned, and reassigned any number of times until the task is dispatched. This happens when a crew logs on to the shift and the task is dispatched. Tasks may also be advance dispatched to crew shifts and contractor capacities.

The scheduler may also respond to various types of requests. Refer to Request Types for more information.

Request Types

A scheduler that handles requests can handle the following request types:

- **Chooser** – returns a list of recommended shifts or contractor capacities for an activity
- **Appointment Booking** - returns a list of valid possible appointments for an activity.

  Appointment booking refers to how the system handles requests from the host system for appointments and how it responds with available appointment windows.

  The appointment booking process receives appointment requests from the host that specify the time periods for which the scheduler should return a list of available appointment windows. Refer to the Appointment Booking Groups for more information.
- **Conditional Booking** - confirms the schedule of an activity when it is interfaced to the system.
- **Crew Route and Directions** – returns the street level route as well as driving directions for a list of tasks in a crew shift's schedule.

The scheduler uses a scheduling horizon to determine the number of days into the future to consider when scheduling crew shifts and activities. The scheduler uses a long term horizon to determine the number of days into the future to consider for long term requests. The long term horizon must be the same or larger than the scheduling horizon.

A scheduler configured to handle requests can also be restricted to handle short term requests only (i.e. those set within the scheduling horizon), long term requests only or both.

Scheduling Parameters

Scheduling parameters allow you to define how the scheduler functions and how it makes scheduling "decisions". Refer to Overview of Scheduler Parameters and Parameter Descriptions for a complete list of scheduler parameters.

Parameters affecting scheduling may be defined at three levels:

- Task-based parameters are defined directly on activity or at the activity type level. These are also referred to as Entity Parameters in the Parameter Definition portal.
- Shift-based parameters are defined in a shift cost profile or individual crew shift record. These are also referred to as Entity Parameters in the Parameter Definition portal.
- Global scheduler parameters are defined in a **Scheduler Configuration** record. These are set at initial implementation and will likely not change again. Refer to Scheduler Configurations for more information.

Understanding Auto Dispatch

Dispatching is an action that locks a task's sequence in the shift schedule so no further optimization takes place and marks it as ready to be sent to the crew. Dispatching can be either manual (initiated by a dispatcher) or automatic (performed by the system without user initiation).

- Automatic dispatching applies only to a started shift. Until the crew starts their shift, the system does not automatically mark any task as ready to be dispatched.
- Manual dispatching applies to activities only. Breaks and POU tasks are always automatically dispatched.
- A dispatcher may manually request to dispatch an activity at any time. If a crew's shift has not started yet, then the activity remains queued for dispatch until the crew starts the shift.

You can enable or disable automatic dispatching of activities globally or for specific crew shifts or activity types.

Setting Auto Dispatch Globally

To enable or disable auto dispatching globally:

- Set the **Enable Auto Dispatch** parameter to True or False in the scheduler configuration.
  (This is listed under Real Time parameters on the **Scheduler Configuration** portal.)
  - **True** - auto dispatching of activities is enabled globally, but can be disabled at the activity or shift level.
  - **False** - auto dispatching is disabled and cannot be enabled at the shift or activity level.

Additional scheduler parameters affecting auto dispatching include:

- Break Dispatch Mode
- POU Dispatch Mode
- Non-Productive Task Dispatch Mode
- Auto Dispatch Stability Period
- Auto Dispatch Interval
- Auto Dispatch On Completion
- Auto Dispatch Time Horizon
- Emergency Dispatch Mode

Setting Auto Dispatch for Crew Shifts

To enable or disable auto dispatching at the crew shift level, set the following attributes for a particular crew shift or crew shift template:

- **Drip Mode:** Indicates how the system should handle dispatching of activities to the crew shift. Valid options are:

None: Automatic dispatching is disabled. All scheduled activities must be manually dispatched to the crew by the dispatcher or by another system-invoked process.

All: All scheduled activities are automatically dispatched to the crew and any additional activities scheduled throughout the shift are also auto-dispatched. This excludes any activities where the Auto Dispatch attribute is set to No on their Activity Type.

Standard: Only a fixed number of scheduled activities are automatically dispatched to the crew at any given point of time. The fixed number is specified in the Drip Horizon. Note that if an activity's type does not allow automatic dispatching, the activity will not be assigned to shifts that are set to Standard drip mode.

- **Drip Horizon:** Number of activities that can be dispatched to a crew at any time. Valid only if Drip Mode is set to Standard.

Setting Auto Dispatch for Activity Types

To enable or disable auto dispatching for a particular activity type:

- Set the **Auto-Dispatch** attribute at the activity type level.
  - **Yes** - activities of this type will be automatically dispatched to the assigned crew.
  - **No** - activities of this type will not be automatically dispatched, overriding the global or shift-based auto-dispatch settings.

Note: Preventing certain types of activities from being automatically dispatched is applicable only if auto-dispatch is enabled globally and the Drip Mode of the crew shift is set to All. Manual dispatching is not allowed if a shift is being drip fed tasks (Drip Mode of Standard).

Managing the Scheduling Process

The term scheduler refers to the system component that manages a single scheduling process. An implementation typically runs at least two schedulers to provide for backup, but may run additional sets of schedulers if required for scalability purposes.

Each uniquely named scheduler is configured to manage a single scheduling process. Scheduling process responsibilities include:

- Holding - does not schedule shifts or tasks, only captures tasks and shifts that cannot be assigned to any online scheduler. Only create one scheduler of this type.
- Request Handler Only - only handles requests
- Optimization - uses the scheduler's full optimization functionality, but does not handle requests.
- Optimization and Request - handles optimization and requests.

A holding scheduler does not schedule shifts or tasks. It is designed to capture tasks and shifts that cannot be assigned to any online scheduler. One and only one holding scheduler should exist. Any task or shift attached to a holding scheduler is evidence of a scheduling error.

Using Multiple Schedulers to Scale the Scheduling Process

The scheduling process is very complex and has high memory requirements that increase dramatically as the number of tasks being scheduled increases. Memory requirements also increase as the scheduling horizon (the number of days into the future that must be scheduled) increases. To achieve optimum performance without sacrificing scheduling efficiency, the system can be configured to use multiple schedulers at once and limit the number of tasks and shifts for which each scheduler is responsible.

Oracle provides general guidelines for calculating the optimum number of schedulers required for a particular installation:

- Define a variable, **L**, to be the maximum task load that can be scheduled by a single scheduler. 20,000 is the typical maximum.
- Determine the total number of tasks, of unique location, to be scheduled in a given scheduler area within the scheduling horizon.
- If the task count is less than L, then a single scheduler can cover the entire area. One extra scheduler will be required for backup.
- If the task count is greater than L, then the scheduler area must be covered by multiple schedulers working in parallel, and the area must be divided into smaller segments called optimization areas. Each scheduler is typically configured with a single optimization area.
- To calculate the number of optimization areas (schedulers) required for a large task count, use the following guidelines:
    - Though the typical maximum load (L) for a scheduler is about 20,000, schedulers configured with an optimization area should be designed for a smaller average load, typically 62.5% of L, which we call *oL*.
    - The total number of schedulers required for the scheduler area can be estimated as follows, where *n* is equal to the expected maximum task count:

      ((*n*/*o*L rounded up to the next whole number) * 3) - 1

The result is multiplied by 3 because three sets of optimization areas are typically required to manage a scheduler area with a large task count. Subtract 1 because the third optimization set can typically have fewer optimization areas than the first two sets. Please note that these rules are guidelines only; greater or fewer schedulers may be required depending on local geographies and particular scheduling requirements.

The table below shows an approximate number of schedulers required for various task counts, assuming the following base values:

- Maximum task count for a single scheduler: *L* = 20,000
- Maximum task count for a single scheduler linked to an optimization area: *o*L = 12,500 (62.5% of L)
- *n* = Task Count

Task Count (n) Low	Task Count (n) High	Scheduler Count	Explanation
0	20,000	2	If the load is less than the maximum task count, then one scheduler can handle the entire load, with an additional scheduler for backup.
20,000	25,000	5	n = 25,000 n/oL (25,000 / 12,500) rounded up = 2 2 * 3 - 1 = 5

25,000	37,500	8	n = 37,500 n/oL rounded up = 3 3 * 3 – 1 = 8
37,500	50,000	11	n = 50,000 n/oL rounded up = 4 4 * 3 - 1 = 11

Thus, every physical location of an area must be covered by at least two schedulers for small task counts (typically under 20,000) or three schedulers for large task counts (typically over 20,000).

> Note: The actual number of schedulers necessary may vary due to geographic constraints or particular scheduling requirements. Some geographic areas may be too large for a single scheduler to process. Determining the appropriate number of schedulers and defining optimum scheduling areas are essential to achieve maximum scheduling efficiency.

Monitoring Schedulers

The application provides the ability to monitor schedulers using standard batch controls. This allows you to use third party enterprise management tools, if desired, to monitor schedulers as you would with any other batch processing.

Every scheduler must be configured with a unique batch control based on **M1-SM** so that monitoring can be executed. This batch control must be referenced on the scheduler record. For more information on creating these scheduler batch controls, refer to the Framework Administrator Guide under "Defining Batch Controls" and in the Framework Business Process Guide under "Batch Jobs".

Once this is configured you can monitor schedulers using standard processing:

- Use an enterprise scheduler which shows if any scheduler submissions have failed. These can typically restart the scheduler using the tool's console.
    - JMX monitoring is also available for enterprise management consoles such as Oracle Enterprise Manager or via simple JMX clients.
- View diagnostics in the **Batch Control Global View** portal.
- View logs in the **Batch Run Tree**.
- View statuses in the **Scheduler** portal in the **Scheduler List** and in the **Scheduler Runtime Details** zone.

  Note that the **Scheduler** tab on **Activities** shows information about how the scheduler is handling the activity and whether or not any issues are detected. You can also access the **Service Area** portal and review the **Covering Schedulers** zone for information on the schedulers covering that service area.

Locating the Applicable Scheduler for a Task or Shift

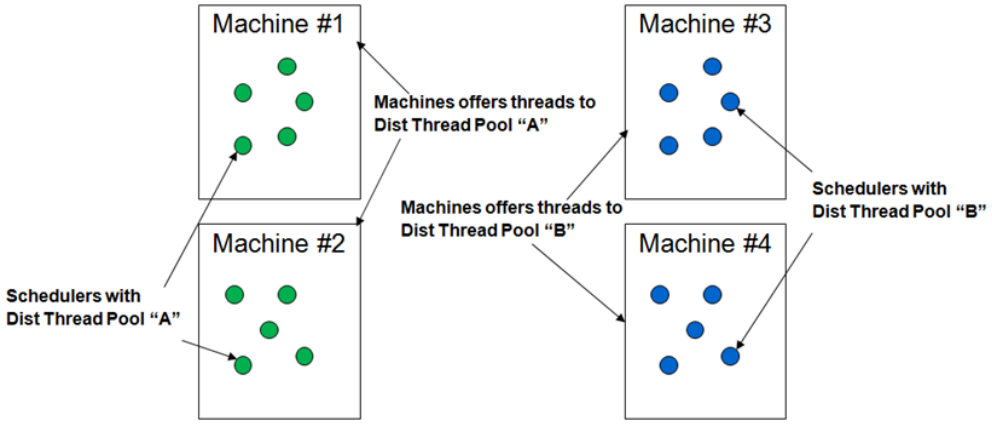
You can configure the system to provide advanced users with the ability to search for and locate schedulers that are handling a particular task or shift. The "Navigate To Scheduler Registry" BPA script can be added to your user's favorites through preferences. This script locates the related scheduler registry if any for the task or shift being displayed.

Failover and Load Balancing

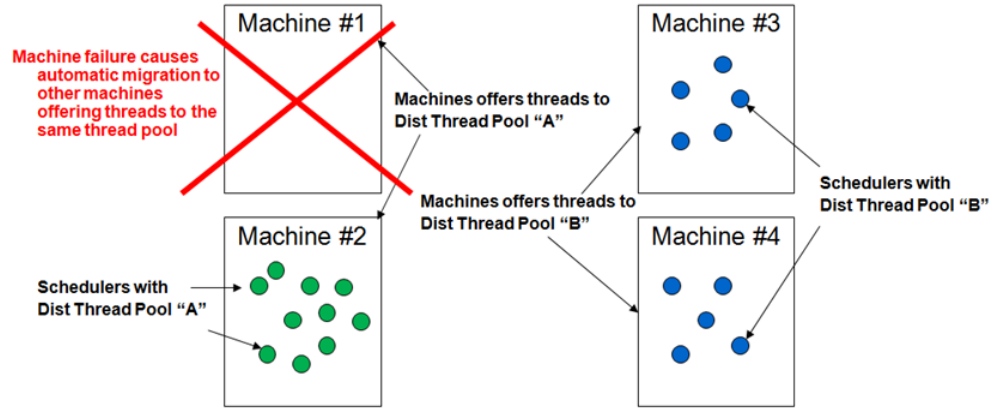
Each scheduler is controlled by a designated batch process. The batch infrastructure provides flexibility around failover and load balancing. Distributed Thread Pools allow a pool of schedulers to be associated to a pool of threads across multiple JVMs on multiple machines.

Load balancing among batch nodes assigns a thread to a JVM with the lowest percentage thread utilization for the requested thread pool. This enables active load balancing where schedulers are spread out among servers.

The following diagram illustrates scheduler distribution between thread pools and machines:



The following diagram illustrates how machine failure causes automatic migration to other machines offering threads to the same thread pool:



Scheduler Areas

A scheduler area defines a geographic area to be scheduled by a number of cooperating schedulers. A scheduler area may comprise one or more service areas. If a scheduler area has too many tasks for a single scheduler, it may be broken down into smaller segments called optimization areas.

Refer to the online help for instructions on creating and maintaining scheduler area records. The following sections provide conceptual information to help you design your scheduler areas before entering them into the system.

Multiple Service Areas within a Scheduler Area

A scheduler is limited to scheduling tasks within its scheduler area. Thus, the set of service areas comprising a scheduler area must at least match the most complex set of service areas covered by any given shift.

Consider a territory with eight service areas: N, E, S, W, NE, SE, NW, SW. Shifts scheduled during the day typically operate in just two service areas, while shifts scheduled during the night can be routed to any of the eight service areas:

- Day Shift (N): N, NE
- Day Shift (E): E, SE
- Day Shift (S): S, SW
- Day Shift (W): W, NW
- Night Shift: N, E, S, W, NE, SE, NW, SW

Since the scheduler area must cover the most complex set of shift service areas, the Night shift is used to model the scheduler area:

Scheduler Area: N, E, S, W, NE, SE, NW, SW.

Using Optimization Areas

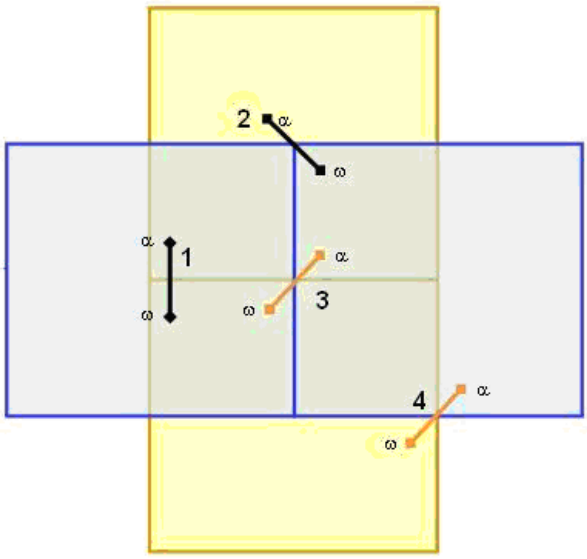
If a scheduler area has too many tasks for a single scheduler to manage (typically over 20,000), the system should be configured to run multiple schedulers in parallel and each scheduler should handle a segment of the total scheduling area. The segments, called optimization areas, should overlap each other. The overlap promotes task/shift migration between schedulers and thus achieves optimal shift routing.

Determining the Best Optimization Area Shape

Every route that crosses an optimization area boundary represents additional workload for the scheduler. The scheduler must acquire all tasks of every shift it schedules, so it is most efficient if every scheduler has most of its shifts wholly within its own optimization area. Thus, the best shape for an optimization area is the one least likely to have routes exiting and entering the area.

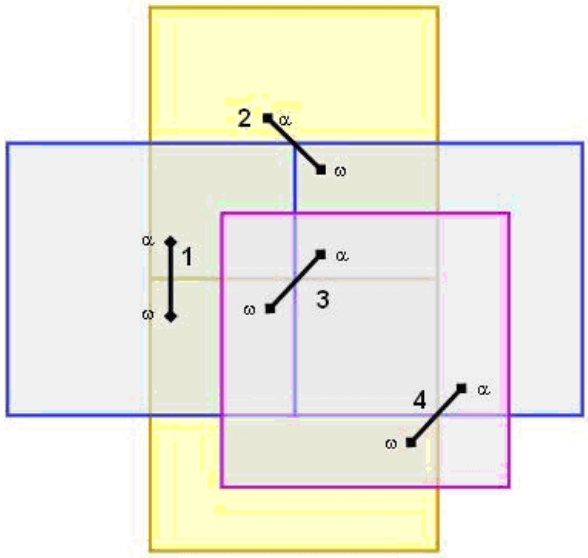
Optimization area shapes must be complementary. Optimization areas of different sets should not share common boundaries and should overlap as much as possible to promote optimum routing.

The diagram below illustrates how a pair of complementary optimization areas may be insufficient for optimal routing at specific points on the map. This simple diagram shows two optimization sets, each with two optimization areas. Each of the four optimization areas shown has its own scheduler - for simplicity, only partial sets are illustrated. The black and orange lines represent four different shifts.



Shifts 1 and 2 can be adequately covered in this configuration, because each one lies entirely within at least one scheduler. Shifts 3 and 4, however, cannot, because no single scheduler contains all the tasks of the shift. This is referred to as the corner problem, which occurs wherever the boundaries of two different optimization areas intersect and the two areas belong to different optimization sets.

Adding a third optimization set, as shown in the following diagram, solves the corner problem. All shifts can be covered as shown, because each one lies entirely within at least one scheduler.



In general, corner problems caused by optimization area overlap must be covered by a third set of optimization areas. If no corner problems exist, only two optimization sets are required.

Follow these general steps when designing optimization sets:

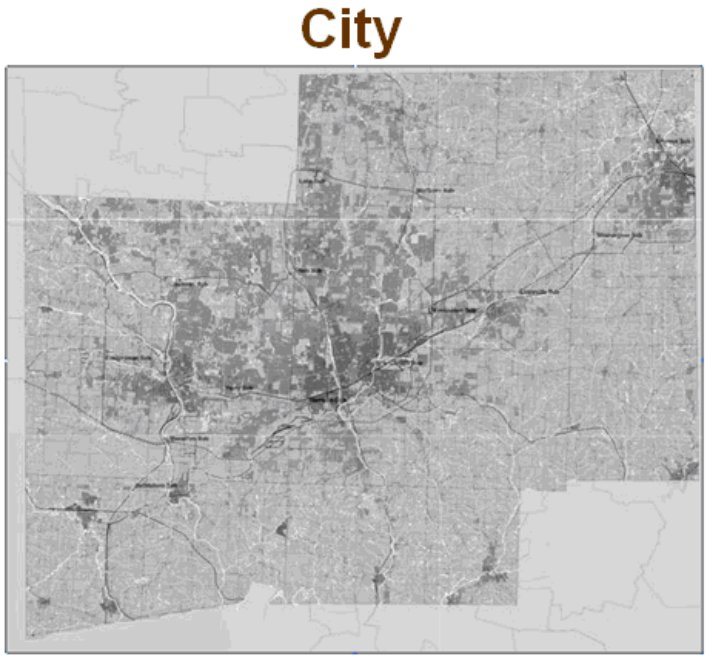
1. Design a set of optimization areas to cover the scheduler area, where each optimization area is as large as possible (the number of tasks bounded by any area should not exceed the

maximum scheduling load plus a growth factor) but still fits into a single scheduler. For best results, always try to use the least number of optimization areas possible and assign only one optimization area to a scheduler (though multiple optimization areas can be assigned to a single scheduler).

2. Design a second set of optimization areas where the optimization area boundaries are as far as possible from the first set's optimization area boundaries.
3. Design a third set of optimization areas to cover any 'corner problems' caused by the overlap of the first two sets.

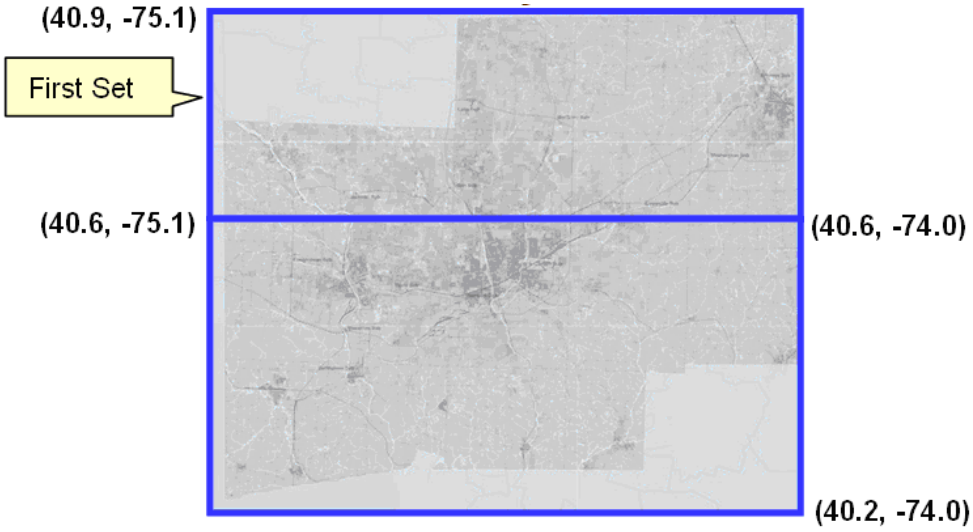
Example:

The following image illustrates a scheduler area, named 'City', that contains more tasks than the physical limit for a single scheduler. Therefore, the scheduler area will require three sets of optimization areas.



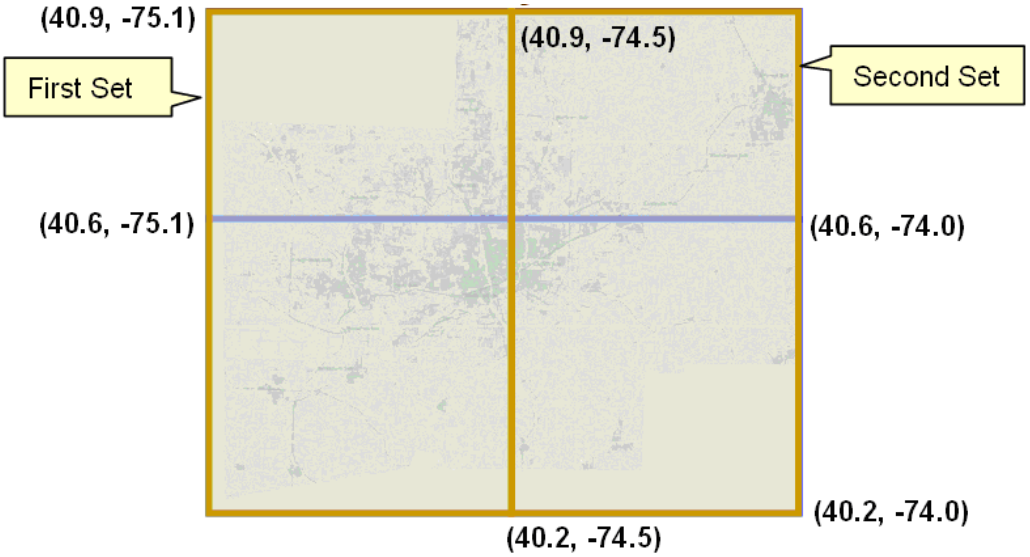
Define the first optimization set with coordinates: (40.9, -75.1 NW) to (40.2, -74.0 SE). The optimization set will have two optimization areas:

- Optimization area 1: (40.9, -75.1 NW) to (40.6, -74.0 SE).
- Optimization area 2: (40.6, -75.1 NW) to (40.2, -74.0 SE).



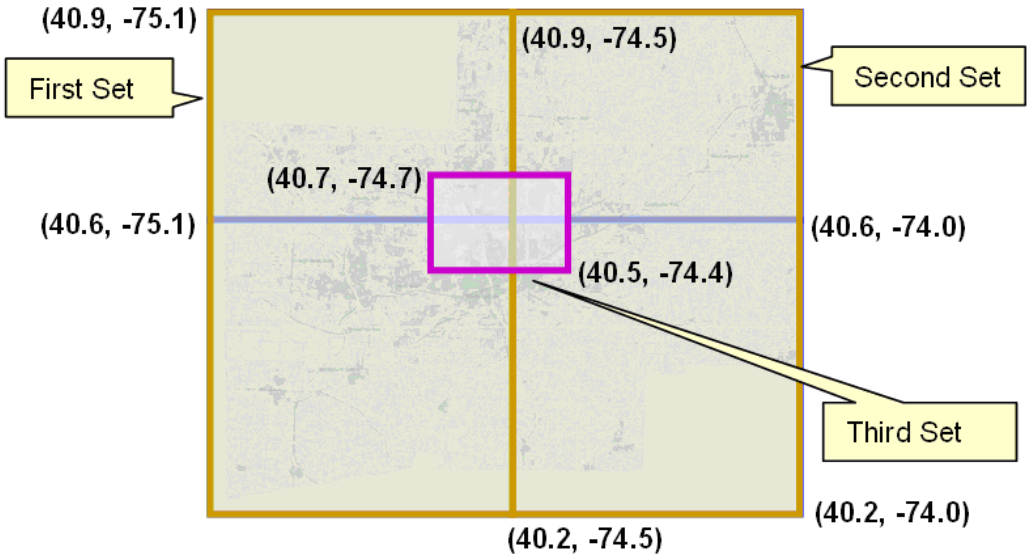
Define the second optimization set with coordinates: (40.9, -75.1 NW) to (40.2, -74.0 SE). The optimization set will have two optimization areas:

- Optimization area 1: (40.9, -75.1 NW) to (40.2, -74.5 SE).
- Optimization area 2: (40.9, -74.5 NW) to (40.2, -74.0 SE).



Define the third optimization set with coordinates: (40.7, -74.7 NW) to (40.5, -74.4 SE). The optimization set will have one optimization area:

- Optimization area 1: (40.7, -74.7 NW) to (40.5, -74.4 SE).



Scheduler Configurations

A scheduler configuration is a predefined set of parameter values that can be associated with one or more schedulers.

Each scheduler is associated with a scheduler configuration that defines the scheduling parameters to be used by that scheduler during the scheduling process.

These parameters include cost controls, performance and optimization settings, and settings to control logging, auto-dispatching, real-time processing and other scheduler functions.

A Scheduler Configuration record references a Cost Profile to define is cost parameters. A cost profile may be referenced by multiple scheduler configurations.

Cost profiles allow you to create pre-defined sets of cost controls that are configured for different circumstances. Users can actively switch between cost profiles to cause the scheduler to respond and optimize work accordingly. For example, an organization can save different profiles based on how conditions change work priorities such as: winter work, summer work, storm conditions, overtime scenarios, etc. Refer to Understanding Cost Control Parameters for more information.

A user with the required authority can switch between profiles so that the scheduler responds and optimizes work accordingly. These changes can be made without shutting down the scheduler.

Users responsible for monitoring schedulers will require training on your business practices associated to when and how to change cost profiles based.

- Use the **Scheduler Configuration** portal to add or change a scheduler configuration or a cost profile. Refer to the online help for instructions on creating a new configuration, completing it, and associating it to specific schedulers.

Using the MapEditor

The MapEditor allows you to create map files used by the scheduler to obtain travel information and to display places and travel paths in a particular geographical region. A scheduler map consists of geographic locations, referred to as nodes, and inter-node links for which the direction, distance, and travel time are known.

When new map updates are sent from HERE maps (formerly Nokia / NAVTEQ), they can sometimes overlay the existing maps and overwrite any edits made by your implementation via the

Map Editor tool. If your system is properly configured, edited maps can be reapplied onto an updated version of HERE maps.

Documentation for this tool is provided in a separate guide. Refer to the Oracle MapEditor User Guide.

Setting Up Schedulers

This section provides a high-level summary of the steps required to set up schedulers and scheduling entities for your organization.

1. Review the scheduler-related plug-ins provided with the base system and determine if any customizations are needed. Refer to the section on Scheduler algorithms in Appendix B: Algorithm Entities for more information.
2. Use the **MapEditor t**o build map files used by the scheduler to create and optimize routes, and calculate travel times and distances. Refer to the MapEditor User's Guide for details.
3. Review the scheduler parameter definitions and adjust the global default values as necessary.
4. Analyze the geographic areas requiring scheduling.
   Each scheduler is configured to cover a specific geographic area called a scheduling area. Use the guidelines presented earlier in this section to determine number of schedulers required. If optimization areas will be used, determine optimal size and shape of each optimization area. Scheduler areas and optimization areas are defined on the Scheduler Area portal.
5. Define various cost profiles to support various circumstances as needed.
6. Determine how many scheduler configurations you must define.

   Note: All schedulers whose scheduler area is associated with the same service area must use the same scheduler configuration. Only schedulers that do not cooperate (that is, they have different service areas) may have different scheduler configurations.
7. For each scheduler configuration, determine the scheduler parameter values to use and the number of days in the future for which data should be considered for scheduling.
8. Identify each scheduler you will be using.
9. Analyze scheduler distribution over servers as well as the necessary thread pools needed to support load balancing and failover.
10. Create a unique Batch Control record based on M1-SM for each scheduler. Ensure to associate the batch control with the appropriate thread pool name as a batch parameter.
11. Create a scheduler record and associate it with its batch control, a scheduler configuration a scheduling area and an optimization area as needed.

Scheduler Registry

The scheduler registry contains a record for each task, shift, appointment booking group, and scheduler configuration that is linked to a scheduler. All objects associated with the scheduling process register themselves to the scheduler registry.

- An activity is registered with all schedulers that host its service area or, if optimization areas are enabled, its optimization areas. If a scheduler's scheduler area is defined using optimization areas, then the activity registers itself to that scheduler only if the scheduler covers the activity location's geocode. If a scheduler is defined using service areas only, then the activity registers itself to the scheduler if it covers its service area.
- A break is registered with all schedulers that host its shift.
- A POU task is registered with all schedulers that have the common services areas with the POU.

- A shift is registered with all schedulers that host any of its service areas.
- Every appointment booking group is registered with all schedulers.
- A scheduler configuration is registered with all schedulers that use that configuration.

The online scheduler monitoring processes update the scheduler registry automatically to remove objects that no longer belong to any of the scheduler's optimization areas (if used) or service areas (if no optimization areas are used) and to add objects that belong to one of the scheduler's optimization areas but are not already linked to the scheduler. This can happen if a scheduler's optimization areas are altered while the scheduler is running.

Posting Actions

The system uses a posting action to communicate information to the scheduler about a change that was made to an entity. The scheduler then communicates this information to its scheduler process.

> **Note**: It is the responsibility of the audit plug-in of the entity's business object to update the registry when changes are made to that entity.

Posting actions in the registry include:

- **Send**: An entity has changed and the entire entity needs to be sent to the scheduler.
- **Send Status**: An entity's state has changed and state-related details for the entity, rather than the entire entity, need to be sent to the scheduler.
- **Send Schedule**: Another scheduler manager has made a schedule change to an entity and schedule-related details need to be sent to the scheduler.
- **Processed**: The registry record has been processed by the scheduler. For example, an object that needed to be sent to the scheduler process was delivered.
- **Remove**: The system has deleted or cancelled an entity (or changed it so that it is no longer relevant to the scheduler) and the entity needs to be removed from the scheduling process (that is, it should no longer be considered for scheduling). After removing the object, the scheduler deletes the registry entry.

If any errors or warnings resulted from the posting action, they appear in the registry record. Depending on your configuration, To Do entries may also be generated for scheduler registry errors.

Scheduler Related Algorithms

Scheduler read algorithms are used to read registered objects and package them for transfer between the scheduler manager and the scheduling process. Read algorithms are enabled for tasks, shifts, appointment booking groups, and scheduler configurations.

The base package provides several algorithms for the following scheduler-related system events. Refer to Appendix B: Algorithm Entities for more information.

Addendum B

Scheduler Cost Controls

Oracle Utilities Mobile Workforce Management and Oracle Real-Time Scheduler are heavily parameterized with the ability to support a myriad of operational use cases and scenarios. Every individual implementation will utilize a unique combination of these parameters to deliver the final operational solution.

This chapter discusses cost control and entity parameters which can be modified to best suit your business practices when configuring the system. Descriptions of the administrative scheduler parameters are provided in Parameter Descriptions. In general these parameters should only be modified with the guidance of your Oracle representative.

For a more complete description of the scheduler in general, refer to Scheduler Configuration.

This chapter includes the following:

Overview of Scheduler Parameters

Scheduler parameters control how a particular scheduler will schedule tasks, optimize routes, and dispatch activities to crews.

Parameters are typically defined as part of the implementation process. The **Parameter Definition** portal allows admin users to set the minimum, maximum, and default values for each scheduler parameter.

Scheduler parameters can be broadly categorized in two ways:

- **System and Administrative** parameters are required to control how the base application operates. As the name suggests, these are applied system wide and control general operating behavior as well as the interaction with external systems through integration. Each individual parameter setting in this group needs to be considered and mapped to the operation to ensure the correct operational behavior.
  System and administrative parameters are associated with a scheduler configuration and their values are maintained on the **Scheduler Configuration** portal. As mentioned, modifications to these parameters can have a very serious impact on the system and require a shut down to commit the changes.

- **Cost Control** and **Entity** parameters are required to provide detailed scheduling policies. These will differ for each implementation depending upon the type of key operations managed by an organization.

  Entity parameters are associated with other objects in the system, such as shifts or activity types, and are maintained on the maintenance portals for those objects.

  Cost control parameters primarily control the day today flexibility and "tuning" of the scheduler solution. These controls can be applied to the production environment without the need to bring down the system.

  These controls are applied to objects that represent the operational structure, such as:

  - **Activities** - data that represents tasks, order or jobs to be scheduled, but which originate in an external HOST system and are passed to the system via integration.
  - **Crew Shifts** - data that represents each working day for each resource that is being scheduled by the system.
  - **Depots** - data that represents locations which can be visited by the scheduled resources, and the operational controls, operational time windows and capacities for those depot locations.

Business Drivers Related to Cost Control Settings

Typically, two major business drivers will guide your configuration of cost control parameters. These business drivers are generally tied to the areas of operation indicated. Of course, any implementation may mix and match these business goals:

- More **Efficient** Service
  Transportation Based Operations
  - Better Leverage of Workforce Skills
  - Reduction of Time Lags / Travel Waste
- More **Effective** Service
  Field Service Based Operations
  - More Responsive to Customer Requests
  - More Assurance of Timely / Complete Response

It is best not to change the Travel Distance and Travel Time costs from their default settings. These act as a reference for all other cost-controls.

Operations Based Parameter Configurations

Most best practices for scheduler configurations fall into one of two main operations areas:

- Field Service Based Operations
- Transportation Based Operations

Field Service Based Operations

Businesses with field service based operations typically require that a person in the field make a physical visit to the customer site. Electric, gas and water utility companies usually fall into this categorization.

Organizations with operations based mainly in the field might focus on the following considerations:

- SLA Adherence
- Days to Serve
- Balance Planned and Reactive Work Orders
- Lowest Resource Schedule

SLA Adherence

Operations that have service level contracts to customers, generally known as service level agreements (SLA), must adhere to certain standards when making service calls. Failure to meet these standards can result in costly penalties and legal consequences.

Cost controls to configure include any lateness and time cost rules from the Shift and Activity groupings.

Sample of Cost Controls to Configure

- Arrival Costs
- Late Cost
- Relative Late Cost (Entity Parameter)
- Resource Attribute Cost
- SLA
  - SLA Flexibility (Entity Parameter)
  - Preferred Time Factor (Entity Parameter)

Days to Serve

Organizations that do not have formal customer service level contracts but that maintain internal targets to offer service within a certain number of days of a customer placing a work order, offer incentives to meet these service goals.

Cost controls to configure include any time dependent cost rules, resource, shift, travel, time and shift promotion rules.

Sample of Cost Controls to Configure

- Late Cost
- Shift Promotion Cost
- Preferred Time Factor
- Complex Activity Span Cost

Balance Planned and Reactive Work Orders

Some scheduler rules can help to balance situations where pre-existing planned work (such as age changes) must be superseded by higher priority reactive work order types (emergency or reconnects or credit restores). The planned work must still be completed to meet service level commitments, while the organization still completes the reactive work tasks.

When planned work orders come into the system, they come in bulk. Reactive work only comes when it occurs, so the scheduler needs to be automated to factor in reactive work. It moves planned work out to make space for reactive work while also factoring in the cost for these changes.

Cost controls to configure include any priority allocation settings, cost rules and time window automatic extension settings.

Sample of Cost Controls to Configure

- Shift Promotion Cost
- Stop Promotion Factor
- Preferred Time Factor
- Late Cost

Lowest Resource Schedule

In organizations where the efficiency of the operation is measured primarily on the utilization of the fewest of the available resources, shift cost and resource rules become very important. When the scheduler is set to consider the lowest resources it attempts to use fewer crews and better resources (such as varying skill levels).

Cost controls to configure include any resource (including contractors), overtime or shift cost rules.

Sample of Cost Controls to Configure

- Resource Attribute Cost
- Overtime Cost
- Shift Cost
- Cost for a Non-Preferred Contractor - This mainly applies for **shift based contractors.** Organizations employing a number of sub-contractors with different agreements, may want the scheduler to prioritize one sub-contractor over another, depending on the Activity. A ranked list of preferred contractors is specified on the activity, any Contractor not specified is deemed to be ranked last.
- Cost for a Non-Preferred Shift Class - This only applies for organizations employing **capacity based contractors**. The same considerations apply as for shift based contractors (as described above), however organizations employing both types of contractors must also configure a "Non Preferred Shift Class" cost.
- Complex Activities require additional configuration to control the elapsed time between the first and last visit, and the duration of visits themselves.
    - Complex Activity Span Cost
    - Under Scheduled Visit Cost
    - Maximum Overlap Between Visits

Transportation Based Operations

Business focused on transportation based operations tend to place a high value on making appointment times and maximizing traveling distances.

Organizations with operations mainly based around transportation might focus on the following considerations:

- Appointment Compliance
- Maximum Density of Appointments
- Zero Over Capacity Schedule
- Lowest Mileage Schedule
- Lowest Resource Schedule

Appointment Compliance

In organizations where appointments are offered to ensure that deliveries and collections are made according to the appointment commitments, lateness time and distance become important factors in the scheduler configuration.

Cost controls to configure include any lateness, time or distance cost rules.

Sample of Cost Controls to Configure

- Late Cost

Maximum Density of Appointments

Most organizations will need to configure the scheduler to ensure that appointments are not scheduled too close together and that there is enough time to complete work in between appointments.

Cost controls to configure include any time or distance cost rules.

Sample of Cost Controls to Configure

- Late Cost
- Apply Minimum Travel Time to All Tasks (Optimizer Behavior Parameter)
- Task Grouping Travel Time (Optimizer Performance)
- Shift Promotion Cost (to promote appointments/fill gaps in earlier shifts)

Zero Over Capacity Schedule

In situations where the work load puts resources over capacity, parameters must be in place to enforce alternate scheduling policies and contingent plans to reallocate priorities.

Cost controls to configure include any weight, volume, capacity, lateness or overtime cost rules.

Sample of Cost Controls to Configure

- Cumulative Capacity 1- 5 Cost
- Depot SLA Window
- Depot Late
- Depot Window Capacity 1 - 5
- Late Cost
- Overtime Cost

Lowest Mileage Schedule

Some organizations place the most value on achieving the highest efficiency in their operations by measuring the economy of scheduled routes. The scheduler should be configured to plot the shortest route possible and within the shortest amount of time.

Cost controls to configure include any time or distance cost rules.

Sample of Cost Controls to Configure

- Late Cost
- Task Grouping Travel Time (Optimizer Performance)

Lowest Resource Schedule

Some organizations place the most value on achieving the highest efficiency in their operations by measuring the utilization of resources. The scheduler should be configured to use the fewest and last expensive number of available resources.

Cost controls to configure include any resource, shift or overtime cost rules.

Sample of Cost Controls to Configure

- Resource Attribute Cost
- Shift Costs - Any parameters related to maximizing shift efficiency.
    - Overtime Cost
    - Service Class Cost
    - Shift Promotion Cost

Understanding Cost Control Parameters

The system uses cost control parameters to encourage or discourage particular actions within the system. Applying cost controls allows the system to produce the most cost-effective schedule while adhering to your business rules and processes.

Cost parameters are applied by defining a cost profile as a type of scheduler configuration then applying that cost profile to the scheduler configuration. During the course of business, schedulers can be re-configured "on the fly" in response to particular conditions (such as changes in weather or overtime requirements) by changing the cost profile on the scheduler configuration. This can be done without stopping the scheduler and the scheduler will immediately respond by optimizing work according to the changed cost profile.

Cost Parameter Types

Cost Parameters are of two types:

- **Relative:** Cost factors that work as multipliers against global costs. For example, consider that an organization has full-time crews and contractors that may be called on to handle excess work. In order to encourage the system to schedule work to contractors only when no full-time crew is available, you could apply a relative cost of ten to contractor shifts, indicating that these shifts would cost ten times as much to use in the schedule as a full-time crew shift. Examples of relative cost parameters are Relative Shift Cost, and Relative Late Cost. Relative costs factors default to a value of 1.0.
- **Global:** Costs that are applied globally at a base level within the schedule. Examples of global cost settings are Late Cost, Shift Cost, Travel Time Cost, and Travel Distance Cost.

Global costs can be of various types:

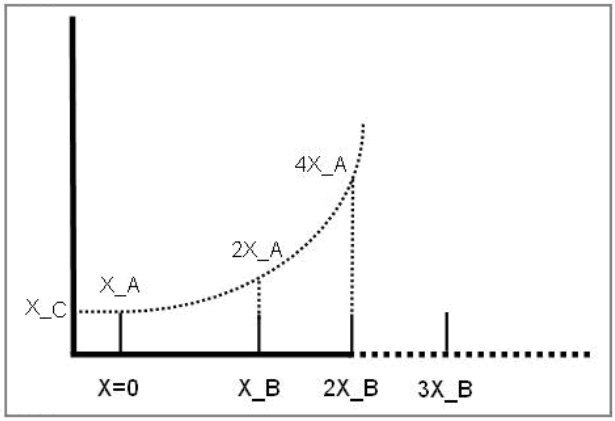
- **Flat:** Costs that are applied once, when the object is activated, and removed when the object is no longer being used. These are also known as “Standing Costs”. The cost for

using this object does not go up over time but stays at a standard or flat cost. For example, the global Shift Cost is the flat cost of utilizing a shift. The Relative Shift Cost parameter can be used as a multiplier against the flat cost. Other examples of flat costs include: Shift cost, cost of a run, cost of goods, hauling costs, site costs, etc.

- **Variable:** Costs that increase linearly based on a measurable variable. These are also known as "Running Costs" or "Variable Costs".
  For example, travel time costs may be expressed as dollars per minute or dollars per hour, and travel distance costs expressed as dollars per mile or dollars per kilometer. Other examples of rate costs include: idle time costs, access window costs, pick up or drop off services, allocation costs, site separation costs, product depot costs, etc.
- **Complex**: Costs that increase exponentially and are nonlinear in nature. The Late Cost parameter, which defines the global cost of late arrival to an activity, is a function of the number of seconds late for arrival. For example, late cost is incurred beginning with the first second the crew is late for arrival. However, the exponential function can be defined so that the late cost is insignificant for the first ten minutes, but starts to ramp up quickly thereafter. This cost pattern models the idea that the customer will accept a crew being 5-10 minutes late, but will be increasingly displeased with any further delays.

  Examples of complex costs include: resource arriving late at a task, overtime, volume, weight, reserve capacity, etc.

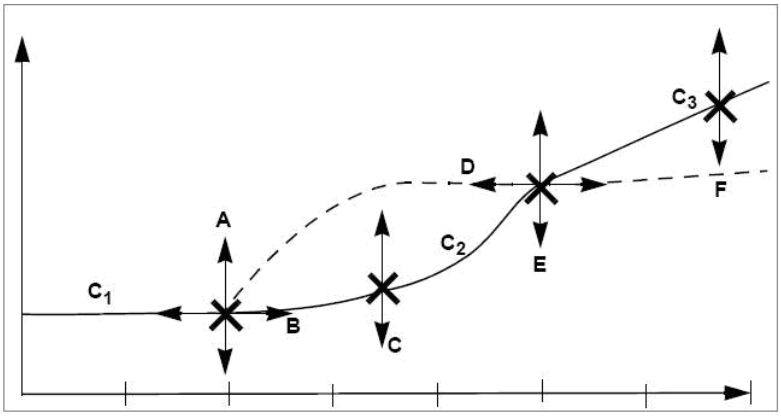
  The following figure illustrates the type of curve represented by a complex cost:



  Complex costs are determined by mathematical functions controlled by a group of three cost control parameters:

  - Cost Value A represents the initial slope for x>=0 (at x=0).
  - Cost Value B represents the doubling rate (the slope of the function doubles every x_B).
  - Cost Value C represents a flat additional cost, creating an initial step.
- **Time-Dependent:** Factors that adjust shift-based costs based on when a shift begins. For example, when the system is scheduling activities several days into the future, the goal is to combine activities to produce the best route for each day. As a result, some shifts may appear under-utilized or even empty. However, when scheduling activities for the current day, the main objective is to add as many activities as possible to fully utilize the crew's time. Time-dependent cost (TDC) factors allow the scheduler to produce the most cost-effective schedule considering these time factors.

To configure time-dependent costs, you must define three distinct curves.



In the diagram above, the horizontal axis is time, in day increments (Day 0, Day 1, Day 2, etc.). The initial curve is always flat. The second curve is either convex or concave. The final curve is always a straight line into infinity.

Six independent variables (TDC Values A-F) are required to define this shape:

- **TDC Value A:** Initial value for the time-dependent cost factor, which is flat from day 0 to TDC Value B. This value must be greater than or equal to zero.
- **TDC Value B:** Controls how long the initial TDC Value A applies, in hours. For example, if the initial factor should apply for all of the first day, then this should be set to 24 (hours).
- **TDC Value C:** Controls the shape of the curve between the initial value and the final slope, referred to as the transition curve. The valid range is 0.1 to 10. A value of 4 produces a curve similar to the solid line in the previous diagram.

  A value of 0.2 produces a curve similar to the dashed line in the previous diagram.

  A value of 1 produces a straight, linear gradient between the initial and final slope, and is a good starting point.
- **TDC Value D:** Number of hours after which the function changes from curved to linear. This value controls the width of the transition curve. This value must be greater than TDC Value B.
- **TDC Value E:** Factor at the point at which the function changes from curved to linear. (This the final factor for the transition curve and the initial factor for the final curve.) This value must be greater than or equal to zero.
- **TDC Value F:** Controls the gradient of the final slope (units/hour).

  Thus, the three curves are defined as follows:
- **Initial curve:** C1 (x:0..B) = A
- **Second curve:** C2 (x:B..D) = ((x - B) / (D - B))C * (E - A) + A
- **Final curve:** C3 (x:D..) = (x - D) * F + E

Time-Dependent Cost Examples

Example 1: TDC Applied to Late Cost

Time-dependent costs can be applied to the **Late Cost** parameter to address a situation where cost should remain the same for the first two days, then rise rapidly 1 cost unit per day over the next two days and then increase by 0.2 units for all the remaining days. In this example, TDC variables would be set as follows:

A = 1

B = 48 (hours)

C = 1

D = 48 (hours)

E = 3

F = 0.0083 (0.2 / 24)

Assuming that 5 minutes of late time is costed at 100 units to begin with, applying these time-dependent costs would result in the following:

For a shift on day 1, the cost is 100

For a shift on day 2, the cost is 100

For a shift on day 3, the cost is 200

For a shift on day 4, the cost is 300

For a shift on day 5, the cost is 320

For a shift on day 6, the cost is 340

For a shift on day 7, the cost is 360

Example 2: TDC Applied to Shift Promotion Cost

Time-dependent costs can be applied to the **Shift Promotion Cost** parameter to promote early completion of activities. The scheduler has no innate preference for activities to be completed as early as possible. Until other constraints are breached, the scheduler is mainly distance driven. Thus, it may schedule an activity for day 5, even though there is capacity on day 1, if it results in less travel. Time-dependent costing can be applied to the Shift Promotion Cost to bias against distance in favor of ensuring that no transport capacity is wasted for day 1. Ideally, costing is set so that day 1 fills up just before it becomes day 0. If costing is too weak, capacity is wasted; if too strong, days fill up well in advance, and the quality of routes suffers.

Example 1: TDC Applied to Reserve Capacity Cost

Time-dependent costs can also be applied to the **Reserve Capacity Cost** parameter to limit the area covered for individual routes when providing appointments. This is done by invoking an additional cost when a vehicle exceeds a certain percentage (such as 50%) of its available time. Thus, the first appointment determines the general direction of that day's run, with the distance travelled for subsequent appointments being restricted through the reserve capacity cost. The more distance is being covered, the stronger the restriction, causing activities that introduce excessive travel to be deferred to another day. Time-dependent costing is required to gradually relax the Reserve Capacity Cost, enabling the system to fill day 0, day l and so on, in case the reserve capacity cost turns out to be too restrictive.

Cost Control Parameter Descriptions

This section provides descriptions of the cost control parameters for reference.

To view these in the system, navigate to **Admin Menu** > **Scheduler** > **Parameter Definition** and select the filter for the **Cost Control** parameter type.

Allocation Cost

Global cost applied to each optional task that is not assigned (i.e., on the free list).

Parameter Name	Default
ALLOCATION	5,000.0000

This parameter encourages the scheduler to fill all unused Crew capacity within the scheduling horizon with optional activities.

Best Practices

- Start with 5000
- Unused Crew capacity that could be filled by optional Activities is not a real problem if this occurs towards the end of the scheduling horizon. If it occurs earlier, and is perceived as affecting the operation, increase the value. Do not set to excessive levels, i.e. beyond 5 x the costs set for skills or resource attributes.

Late Break Cost

Exponential cost for taking a break late.

Parameter Name	Default
BREAK_LATE	• Cost Value a: 1.0000 • Cost Value b: 3600.0000 • Cost Value c: 0.0000

Note that it is not a good idea to use break late costs because they can sometimes inadvertently cause the scheduler to give breaks a higher priority than other considerations. This should be used very carefully and with guidance from an Oracle representative.

Cumulative Capacity 1- 5 Cost

The cost of cumulative on-board capacity of the shift assigned to the activity.

Parameter Name	Default
CCAP1 - 5	• Cost Value a: 0.0000 • Cost Value b: 0.0200 • Cost Value c: 0.0000

Complex Activity Span Cost

Defines costs as a function of the number of days that a complex activity visit exceeds the number of days allocated for completion.

Parameter Name	Default
COMPLEX_ACT_SPAN	• a. 5,000.0000 • b. 2.0000 • c. 0

Best Practices

- Start with 5000
- If CA's generally exceed their allotted span-time by too much, increase a.

Cost for a Non-Preferred Contractor

This cost biases the scheduler towards scheduling an activity on a shift or capacity belonging to a preferred contractor, if specified. This mainly applies for shift based contractors.

Parameter Name	Default
CONTRACTOR_PREFHAVE	3,000.0000

Depot SLA Window

Sets a value to indicate the cost of working a job outside the timeframe of the Depot SLA Window.

Parameter Name	Default
DEPOT_SLA_WINDOW	5,000.0000

Depot Late

Cost as a function of the number of seconds late at a stop. In order to maintain balance with STOP_LATE, the resulting cost is multiplied by the number of jobs transferred here.

Parameter Name	Default
DEPOTLATE	• Cost Value a: 2.0000 • Cost Value b: 1800.0000 • Cost Value c: 0.0000

Depot Window Capacity 1 - 5

Sets a value to indicate the cost of exceeding the capacity limit for a depot time window.

Parameter Name	Default
DEPWINCCAP_1 - 5	• Cost Value a: 10,000.0000 • Cost Value b: 0.0200 • Cost Value c: 0.0000

Shift Cost

Global cost of utilizing a shift. The Relative Shift Cost parameter, defined on the shift cost profile, can be used as a multiplier against this cost.

Parameter Name	Default
HAUL	0.0000

Best Practices

- Start with 0
- If there is a cost associated with sending a crew out, shift cost may be used to minimize the number of shifts the scheduler assigns work to. Generally this means a small increase in distance travelled/activity and overtime for the shifts still being used. If used, shift cost should be in a similar range as attribute and skill costs.

Idle Time Cost

Global cost of a crew being idle.

Parameter Name	Default
IDLETIME	0.0000

Job Attribute

Sets a value for the cost when a Job Preference and Job Attribute do not match. For example, when transporting prisoners, it is preferable not to have hardened criminals and juveniles in the same vehicle.

Parameter Name	Default
JOB_ATTR	5000.0000

Tasks Limit

Exponential cost for exceeding this number of tasks assigned to a shift. The input to this cost function is the number of tasks exceeding the tasks limit. For example:

- If a shift has a maximum tasks limit of 10 and the sum of all scheduled tasks is 12, then the cost input is 2.
- If a shift has a maximum tasks limit of 100 and the sum of all scheduled tasks is 102, then the cost input is also 2, so the resulting cost is the same.

Parameter Name	Default
JOBS_LIMIT	• Cost Value a: 5000.0000 • Cost Value b: 10.0000 • Cost Value c: 0.0000

Over Skill Cost

Flat cost applied for each crew skill level in excess of the required skill level for an assigned task. This is used to discourage the scheduler from assigning crews that are overskilled (either in quality or in quantity) for tasks.

Parameter Name	Default
OVER_SKILL	1,000.0000

Life Span

Sets a value to indicate the cost for late delivery. Cost is represented as a function of the number of seconds a product is on board longer than the maximum- Runlength defined for a Shift. For distribution runs, if a Shift specifies maxrunlength, and an Activity Lifespan, the shortest duration is applied.

Parameter Name	Default
LIFESPAN	• Cost Value a: 1.0000 • Cost Value b: 3600.0000 • Cost Value c: 0.0000

Overtime Cost

Exponential cost as a function of the number of seconds of over-time for a crew's shift. The Relative Overtime Cost parameter, defined on the shift cost profile, can be used as a multiplier against this cost.

Parameter Name	Default
OVERTIME	• Cost Value a: 2.0000 • Cost Value b: 1800.0000 • Cost Value c: 0.0000

Arrival Costs

Exponential cost to advance the arrival of an activity within its time window. This is calculated as a ratio of the arrival into the applicable Arrival Time Window.

- A=Multiplier. Positive values bias towards early arrival, negative towards late arrival.
- B=The ratio of the time window at which the value that A is multiplied by is doubled.
- C=Initial step value applied only once at start.

Parameter Name	Default
PKUP_SERVICE	• Cost Value a: 1000.0000 • Cost Value b: 1.0000 • Cost Value c: 0.0000

Reserve Capacity Cost

Exponential cost for exceeding shift's reserve capacity. Reserve Capacity reserves a percentage of shift-time for the on-site time of activities of specified service classes. The system adds together all on-site time for non-reserved activities and all travel-time (whether for reserved activities or not). If this total exceeds the un-reserved part of the shift time, the Reserve Capacity Cost function calculates a cost from the difference.

Parameter Name	Default
RESERVE_CAPACITY	• Cost Value a: 1.0000 • Cost Value b: 3600.0000 • Cost Value c: 0.0000

Shift Area Cost

Cost for the geographical size of the area covered by a shift's schedule of tasks. Promotes geographical clustering of tasks in a shift. In order to determine the size of a shift's currently scheduled area coverage, the scheduler draws a rectangle encompassing all the tasks associated with the shift and then calculates the length of the diagonal. This Shift Area Cost is applied per meter of the length of this diagonal.

Parameter Name	Default
ROUND	0.0000

Shift Area Time Cost

Cost of time spent from the first to last task of a shift. Promotes geographical clustering of tasks in a shift.

Parameter Name	Default
ROUND_LENGTH	0.0000

No Task In Service Area Cost

Flat cost for each shift that has no tasks within the shift's covered service areas. This cost is calculated in conjunction with the Service Area Cost parameter.

Parameter Name	Default
ROUND_ZONE	0

Resource Attribute Cost

Cost of having a task on a shift where the task's resource-related attributes do not match those on the shift. The cost is applied once for each task with a conflicting attribute.

Parameter Name	Default
RSRC_ATTR	1,000.0000

Depot Run Cost

Cost of activating a run.

Parameter Name	Default
RUN	1,000.0000

Depot Run Separation

Applies a cost to loading goods during a Distribution-run. cost = fABC(l/c) * r;

- l = load added at Stop
- c = capacity available on Resource
- r = remaining Stops within the run

In general this parameter minimizes having load for multiple runs concurrently on board: Main example is to minimize (but not prevent) collecting return goods before the last delivery of the day has been made.

Parameter Name	Default
RUN_SEPARATION	• Cost Value a: 10,000.0000 • Cost Value b: 0.0200 • Cost Value c: 0.0000

Service Class Cost

Cost of having a task on a shift where the task does not belong to any of the service classes allowed to be performed by the shift. The cost is applied once for each task with a conflicting attribute.

Parameter Name	Default
SERVICE_CLASS	1,000.0000

Shift Promotion Cost

(separate from Shift cost)

Global cost of promoting a shift. This allows a preference to be applied for scheduling tasks to earlier shifts. The task-based Relative Shift Promotion Cost parameter, defined on activity type, can be used as a multiplier against this cost. TDC MUST be used. Default: 0, 0, 1, 48, 4, 0.042

Parameter Name	Default
SHIFT_PROMOTION	500.0000

This parameter allows you to apply variable costs to have work on a shift scheduled as close to "now" as possible. The parameter links with time dependent cost rules so that the cost changes over time. For example, you configure parameters so that the wait time for an appointment is considered as more expensive. Appointments that are scheduled immediately are less costly whereas, the further from "now" the appointment is scheduled, the more costly it becomes.

Best Practices

- Start with 500
- If Activity Priority is not sufficiently adhered to, increase. Note that "priority" is costed. The "priority cost" is traded off against other costs like SLA and distance travelled. Therefore strict ordering by priority is not guaranteed, indeed unlikely.

Cost for a Non-Preferred Shift Class

This cost biases the scheduler towards scheduling an activity on a shift or capacity, depending on which was specified as preferred. This only applies for organizations employing capacity based contractors.

Parameter Name	Default
SHIFTCLASS_PREFHAVE	3,000.0000

Best Practices

- Start with 3000
- Non-Preferred Shift Class may be used to "prefer" Capacity over Shift, or vice versa. Increase the value if the scheduler is not sufficiently adhering to the preferences specified on the activities. Lower the value if "Shift Preferred" activities remain scheduled to shifts despite introducing excessive travel or overtime.

Site Cost

Cost for non-sequential visits made to the same site. Promotes working of all activities at the same site by a single crew in a single visit.

Parameter Name	Default
SITE	0.0000

Stop Promotion Factor

Controls prioritization of activities within a Shift.

Parameter Name	Default
STOP_PROMOTION_FACTOR	0.2000

Best Practices

- Start with 0.2
- If priority within a Shift is not an issue, set to 0.0 to obtain the best routes.

SLA Window Cost

Flat cost of working an activity outside its SLA time window.

Parameter Name	Default
STOP_SLA_WINDOW	0.0000

Late Cost

Global cost of late arrival to an activity. This exponential cost is a function of the number of seconds late for arrival.

The Relative Late Cost parameter (defined at the task level) can be used as a multiplier against this cost.

Parameter Name	Default
STOPLATE	• Cost Value a: 2.0000 • Cost Value b: 1,800.0000 • Cost Value c: 0.0000

Late cost applied with the example allocation cost settings would drive the orders to the front of the queue globally. It is a property of the job time window.

Travel Distance Cost

Global cost of time spent travelling, includes time contributions by site delay. The Relative Travel Time Cost parameter, defined on the shift cost profile, can be used as a multiplier against this cost.

Parameter Name	Default
TRAVELDIST	0.1000

Travel Time Cost

Global cost of time spent travelling, includes time contributions by site delay. The Relative Travel Time Cost parameter, defined on the shift cost profile, can be used as a multiplier against this cost.

Parameter Name	Default
TRAVELTIME	1.0000

Under Scheduled Visit Cost

Cost as a function of the number of seconds that a Complex Activity's Visit Duration is under the Target Visit Duration (Where Target Visit Duration will typically be the Average Visit Duration except for the last Visit where the Target Visit Duration will be the Fixed duration as assigned by the scheduler).

Parameter Name	Default
UNDER_SCHED_VISIT	• 1.0000 • 1,800.0000 • 0.0000

Best Practices

- Start with 1000
- Increase if the scheduler adds too much work before and/or after Visits. First ensure the TVD is set correctly.

Wait Time Cost

Quadratic cost for the shift being idle. The shift-based Cost Idle parameter, defined on the shift cost profile, can be used to control the cost on a shift-by-shift basis.

Parameter Name	Default
WAIT_TIME	0.0000

Wait Time Past Cost

Quadratic cost for shift being idle in the past.

Parameter Name	Default
WAIT_TIME_PAST	0.0000

Window Cost

Global cost of using a task's arrival time window. The Relative Window Cost parameter (defined in the task time window) can be used as a multiplier against this cost.

Parameter Name	Default
WINDOW	0.0000

Service Area Cost

Flat cost for each task outside the shift's preferred service areas. This is used to encourage crews to work in their preferred service areas.

Parameter Name	Default
ZONE	1000.0000

Entity Parameters

Entity parameters are generally configured on the activity type, the shift cost profile, on the task or other specific entity it relates to. The configuration location is specified in the description of each parameter.

To view these in the system, navigate to **Admin Menu** > **Scheduler** > **Parameter Definition** and select the filter for the **Entity** parameter type.

Cost Idle

The parameter is defined on the shift cost profile.

If checked (True), the scheduler considers the cost of idle time for a specific shift's crew, weighted according to the "time from now" factor. No cost is applied if the idleness accrues further than the wait horizon time interval and, within this horizon, the applied cost is stronger if the idleness is closer to the current time.

If unchecked (False), idle time cost is not considered.

Parameter Name	Default
COST_WAIT_SHIFT	True

Shift Distance Cost

Shift cost profile factor used as a multiplier against the global Travel Time Cost. For example, a relative distance cost of 3 means that the distance cost for a particular shift is three times more than the distance cost defined globally.

Parameter Name	Default
DIST_COST	1.0000

Relative Late Cost

Relative cost of arriving late to this activity. For example, a relative late cost of 3 means that the cost of late arrival for a particular activity is three times more than the late cost defined globally. This is used to discourage scheduling activities with planned late arrival time. This is defined at the task level.

Parameter Name	Default
LATE_COST	1.0000

Relative Overtime Cost

Shift cost profile factor used as a multiplier against the global Overtime Cost parameter. For example, a relative overtime cost of 3 means that overtime for a particular shift is three times more than the overtime cost defined globally.

Parameter Name	Default
OVERTIME_COST	1.0000

Relative Efficiency

Defines a mobile worker's efficiency relative to expected efficiency for mobile workers of this type. For example, if a mobile worker is twice as efficient as other mobile workers of this type, the Relative Efficiency is 2. A crew shift's relative efficiency is the average of all its allocated mobile workers' relative efficiencies.

Parameter Name	Default
REL_EFFICIENCY	1.0000

Relative Speed

Defines a vehicle's speed relative to the expected speed for vehicles of this type. For example, if a vehicle is twice as fast as the expected speed for vehicles of its type, the Relative Speed is 2. A crew shift's relative speed is the lowest of all its allocated vehicles' relative speeds.

Parameter Name	Default
RELSPEED	1.0000

Relative Shift Cost

Shift cost profile factor used as a multiplier against the global Shift Cost parameter. For example, a Relative Shift Cost of 3 means that the cost of activating the shift is three times more than the shift cost defined globally.

Parameter Name	Default
SHIFT_COST	1.0000

SLA Flexibility

Period of time before and after the preferred time window (PTW) within which an activity must be scheduled to avoid incurring added cost. This is defined on the activity.

If arrival is outside this time period, the scheduler applies a fixed maximum cost (specified in the Preferred Time Factor field for the activity type). For example, if the SLA Flexibility is set to 20 minutes, then scheduled arrival must be no more than 20 minutes before the start of this preferred time window or 20 minutes after the end of this window; otherwise, the Preferred Time Factor cost is applied.

Format: hhmmss.

Parameter Name	Default
SLA_FLEXIBILITY	60.0000

Preferred Time Factor

Fixed cost that will be applied for working an activity outside SLA Flexibility period. The cost is applied when the job is done. This is defined on the activity type and sets the cost factor for working an activity outside of the preferred time window.

Parameter Name	Default
SLA_PRIORITY	1.0000

Relative Shift Promotion Cost

Task-based cost factor used as a multiplier against the global Shift Promotion Cost. A higher value puts more pressure on the scheduler to find a cheaper (i.e., earlier) shift for tasks of this type. This value is defined at the activity type level and affects activity priority.

Parameter Name	Default
TASK_SHIFT_PROMOTION	1.0000

Relative Travel Time Cost

Shift cost profile factor associated with travel time for a crew shift. This is used as a multiplier against the global Travel Time Cost. For example, a Relative Travel Time Cost of 3 means that the cost of travel time for a particular shift is three times more than the travel time cost defined globally.

Parameter Name	Default
TIME_COST	1.0000

Relative Window Cost

This task-based cost factor is used as a multiplier against the global Window Cost. It defines the preference of an activity's time window relative to other time windows. For example, a Relative Window Cost of 3 means that the cost of using an activity's particular time window is three times more than the Window Cost defined globally.

Parameter Name	Default
WINDOW_COST	1.0000

Addendum C

Scheduler Parameters

This chapter provides descriptions of the administrative scheduler parameters. Cost control and entity parameters are described in Understanding Cost Control Parameters. These administrative parameters should only be modified with very careful consideration and with the guidance of you Oracle representative.

Parameter Descriptions

This section provides descriptions of the administrative scheduler parameters for reference.

Navigate to **Admin Menu** > **Scheduler** > **Parameter Definition** and select the filter for the appropriate parameter type:

Parameter Type	Description
Appointment Generation Parameters	Controls how appointment booking requests from the host are processed.
Connection Parameters	Controls how the scheduler sends and receives data to/from client applications.
Understanding Cost Control Parameters	Defines how to calculate and apply costs used to build the most cost-effective schedule.
Entity Parameters	Defines shift-based and task-based parameters that are applied against global parameters.
General Administration Parameters	Defines system-wide scheduling parameters.
Logging Parameters	Controls how and where the scheduler creates log files.
Map Configuration Parameters	Defines mapping parameters used to build and optimize routes.
Optimizer Behavior Parameters	Parameters that affect optimization of the scheduler.
Optimizer Performance Parameters	Defines parameters that affect system performance.
Real Time Parameters	Controls the scheduler's real-time functionality.
Reference Time Parameters	Controls how and when reference time is updated.

Parameter Type	Description
Scheduler Manager Parameters	Defines parameters used by scheduler manager functions.
Site Parameters	Defines the default Coordinated Universal Time and the error translation file used to translate error messages at initialization.

Appointment Generation Parameters

Use appointment generation parameters to configure settings for how appointments are automatically generated by the scheduler based on various criteria such as the available appointment windows, cost limits, and filtering.

Chronological Filtering

If checked (True), non-chronological appointment windows are removed from (filtered out of) the returned list of appointment windows. If unchecked (False), non-chronological filtering is disabled.

This parameter configures the Slot Generator so that it does not return higher cost appointment slots which are earlier than later less expensive appointment slots.

Some businesses sort the appointment slots returned by date and time and tend to select the earliest slot, while disregarding the cost of the slot. This results in less efficient schedules and tends to produce unfavorable performance metrics.

Parameter Name	Default
slotGenChronologicalFilter	False

Chronological Filtering Tolerance

Maximum number of hours an appointment window can start before the previous one. Applicable only when the chronological filtering is enabled.

This parameter configures the Slot Generator so that it does not return higher cost appointment slots which are earlier than later less expensive appointment slots.

Some businesses sort the appointment slots returned by date and time and tend to select the earliest slot, while disregarding the cost of the slot. This results in less efficient schedules and tends to produce unfavorable performance metrics.

Parameter Name	Default
slotGenChronoFilterTol	12

Termination Threshold

Sets the number of appointment slot candidates that will be costed after the cost-limit has been reached during slot generation. During the slot generation process the slot generator may come across a group of slots that return a value greater than the MAX_UNITS specified in the CHS_SLOT message.

This parameter sets the number of consecutive replies above MAX_UNITS that will be processed before the slot generator terminates.

Parameter Name	Default
slotGenTermination	30

Window Candidate Minimum

Minimum number of candidates considered for a single time window by the appointment booking request process.

Parameter Name	Default
slotGenWinCandMin	10

Window Filtering

Controls filtering of returned appointment windows based on time window border conditions.

Contained within: An appointment window must be within the request time window.

Starts within: An appointment window must start within the request time window.

Ends in: An appointment window must end within the request time window.

Overlaps: An appointment window must overlap with the request time window.

Note that these conditions apply on top of the initial selection criteria indicating that the task can be scheduled to be worked at some time during the appointment window, based on the availability of suitable crew and task time window.

The filters are explained in the following example:

A request was sent with 8am to 10pm time window and the slots are specified as 7-12, 14-19, 19-23.

- • For filter (SG_CONTAINED_IN) 14-19 slot will be considered
- • For filter (SG_STARTS_IN) 14-19 and 19-23 slots will be considered
- • For filter (SG_ENDS_IN) 7-12, 14-19 slots will be considered
- • For filter (SG_OVERLAPS) 7-12, 14-19, 19-23 slots will be considered

Parameter Name	Default
slotGenWinFilter	SG_CONTAINED_IN

Connection Parameters

Connection parameters control how the scheduler sends and receives data to/from client applications.

Client Timeout

Inactivity timeout interval. After this number of seconds of no activity, the scheduler will drop the connection to a client. Set to -1 to disable timeout.

This is set internally and should not be changed.

Parameter Name	Default
clientTimeout	-1

IO Buffer Size

Size (in kilobytes) of the data buffer used by IPC sockets for sending and receiving data. Used to set a sufficient buffer size without wasting memory resources. This is set internally and should not be changed.

Parameter Name	Default
io_bufsize	128

Maximum Packets

Maximum number of packets that will be read in one socket poll. This should not be changed.

Parameter Name	Default
maxPackets	1000

Maximum Wait Block

Maximum time (in seconds) to wait on a blocked socket before closing the socket.

Parameter Name	Default
maxWaitBlock	300

Poll Time Interval

Time interval at which sockets are polled for incoming data. This value should not be changed.

Used to set the optimum rate of communication between processes.

Parameter Name	Default
pollTimeInterval	0.1

Server Mode

If True, sets the mode of operation to Server and Client. In this mode, the scheduler immediately grants connection to all requesting clients.

If False, the scheduler first obtains a connection from the server process and only then grants connection to all requesting clients.

Parameter Name	Default
serverMode	False

Socket Buffer Size

Internal socket buffer size for sending data. This is the maximum amount of memory (in kilobytes) that can be allocated per socket. It should be set to at least twice the size of the schedule.

This value should be set to at least twice the size of the schedule. The schedule size can be determined by saving to a '.hip' file.

Parameter Name	Default
socketBufferSize	20000

General Administration Parameters

General administration parameters control system-wide scheduling parameters.

Test Level

Refers to additional code for performing run-time consistency checks.

0: no optional tests
1: consistency tests that have a negligible impact on performance
3: consistency tests that have some impact on performance
5: consistency tests that may significantly alter performance

Normal setting is 0 and should not be changed in a production environment.

Parameter Name	Default
testLevel	0

Logging Parameters

Logging parameters control how and where the scheduler creates log files.

Auto Save Directory

Directory path name in which auto save files are stored.

Parameter Name	Default
autosaveDirectory	AutoSave

Auto Save Files

Number of auto save files to store.

Parameter Name	Default
autosaveFiles	20

Auto Save Interval

Interval (in minutes) between each autosave. When set to zero, auto save is disabled.

Parameter Name	Default
autosaveInterval	0

Backup Auto Save Directory

Directory path name in which backup auto save files are stored.

Parameter Name	Default
dualAutosaveDirectory	0

Log Compression

File compression method for log files. Compression of log files is available only for UNIX installations. The default is gzip.

Parameter Name	Default
logCompress	gzip

Log File

Path and prefix of log files. Must be a valid and accessible location.

Parameter Name	Default
logFilePrefix_MSG	LOG\smauto

Log Hours

Number of hours each log file spans. Log files are usually sent as e-mail attachments. Setting this value to 1 hour will keep file size to a minimum.

Parameter Name	Default
logHours_MSG	1

Log Level

Level of detail that will be logged for the process. Log levels:

0=No logging

1=Logs real-time/operator events/status changes, shift/ task status changes, etc.

2=Logs operation of automated processes impacting schedule execution, dead-reckoning, auto dispatch, etc.

3=Operation of optimizer

4=Anything else

5=Spare

6-9=Same as levels 1-4, but accompanied by performance penalties or excessive output

Parameter Name	Default
logLevel	3

Log to File

If checked (True), saves all log messages to a file. This should always be True.

Parameter Name	Default
logToFile_MSG	TRUE

No Log to Standard Output

If True, logging to standard output is disabled.

If False, logging to standard output is enabled.

Parameter Name	Default
noLogStdOut	TRUE

Number of Log Files

Number of log files stored.

Parameter Name	Default
numLogFiles_MSG	168

Received HIP Logging Flag

If checked (True), all the HIP packets received from the scheduler are logged.

If unchecked (False), received packets are not logged.

Parameter Name	Default
rxHipLogging	TRUE

Memory Logging

Enables memory logging within the scheduler

If checked (True), all are logged.

If unchecked (False), memory is not logged.

Parameter Name	Default
shMallocMemLog	FALSE

Transmitted HIP Logging Flag

If checked (True), all HIP packets sent from the scheduler are logged.

If unchecked (False), sent packets are not logged.

Parameter Name	Default
txHipLogging	TRUE

Map Configuration Parameters

Map configuration parameters are used to build and optimize routes.

Log Map

If checked (True), the system logs the number of map nodes, links, and matrix rows loaded.

If unchecked (False), these items are not logged.

Parameter Name	Default
logMap	FALSE

Map File

Directory path and filename of the connectivity map for use by the scheduler. A map consists of a list of nodes (geographical locations, suburbs, etc.) and a list of inter-node links (travel paths) whose direction, distance, and travel time are known. This data is used to create the schedule.

Parameter Name	Default
map	0

Matrix Distance Factor

Distance factor (integer multiplier) used in the objective function to calculate the best route. If this is set to zero, the fastest routes are calculated. If Matrix Time Factor matrixTimeFactor is set to zero, the shortest routes are calculated. The contribution of distance and time are approximately equal when the ratio of Matrix Time Factor / Matrix Distance Factor matrixTimeFactor / matrixDistFactor is approximately 20. Do not change unless advised by Oracle.

Parameter Name	Default
matrixDistFactor	1

Matrix Segmentation

If checked (True), segmentation matrix is generated only for tasks that are compatible with the shifts.

If unchecked (False), segmentation matrix will be generated for all tasks in the system.

Parameter Name	Default
matrixSegmentation	TRUE

Matrix Time Factor

Time factor (integer multiplier) used in the objective function to calculate the best route. If this is set to zero, the shortest routes are calculated. If Matrix Distance Factor is set to zero, the fastest routes are calculated. The contribution of distance and time are approximately equal when the ratio of Matrix Time Factor / Matrix Distance Factor matrixTimeFactor / matrixDistFactor is approximately 20. Do not change unless advised by Oracle.

Parameter Name	Default
matrixTimeFactor	40

Matrix Toll Factor

Toll factor (integer multiplier) used in the objective function to calculate the best route. Used to discourage specific roads or road-segments.

Parameter Name	Default
matrixTollFactor	0

Matrix Update Interval

Interval (in seconds) to check the matrix for unused cells.

Parameter Name	Default
matrixUpdateInterval	900

Matrix Version

Matrix implementation version. Options include:

- Full matrix, no segmentation
- High connectivity
- Low connectivity
- Sparse matrix, much segmentation
- Sparse matrix, symmetrical segmentation

The default is set by Oracle for best map performance and should not be changed.

Parameter Name	Default
matrixVersion	LOCONN

Off Map Speed

Used if the specified coordinates do not match the road-network. This is the travel speed (meters/per second) between a task location and the nearest road or other task location.

Parameter Name	Default
offMapSpeed	20

Route Cache Memory Limit

Maximum amount of memory the route cache will use.

Parameter Name	Default
routeCacheMemoryLimit	10

Warning Distance

Sets the maximum distance that task locations are allowed to be from the nearest map road before a warning is issued.

Parameter Name	Default
warnDist	5000

Optimizer Behavior Parameters

Optimizer behavior parameters affect how the scheduler creates the most efficient and effective travel routs, arrival times, routes, etc.

Apply Minimum Travel Time to All Tasks

If checked (True), minimum travel time between tasks is applied for all tasks.

If checked (False), minimum travel time is applied between activity tasks only.

Parameter Name	Default
applyMinTravelTimeToNonStop	FALSE

Arrival Margin

Desired number of minutes prior to the start of the time window for arriving at a task.

Parameter Name	Default
arrivalMargin	0

Expected Average Work Rate of Visits

Used by the scheduler to estimate completion time for a complex active for which not all visits have been scheduled.

Parameter Name	Default
avgVisitsPerDayForMST	1.0

Unexpected Event Handling

If checked (True), the system automatically identifies and resolves issues caused by unexpected events. For instance, a long duration task may be unexpectedly allocated to a shift, introducing 3 hours of overtime. The system will then identify activities that may be transferred to other shifts, in order to relieve overtime.

If unchecked (False), this feature is disabled.

Parameter Name	Default
bUEC	FALSE

Cost by Round

Determines how round-based costs are bounded. Set to TRUE to have round costs bounded by DepotStops. Set to FALSE to have round costs bounded by LogStops. "Round" based costs are usually calculated on a depot-based run, but may optionally be calculated on an entire Shift.

Parameter Name	Default
costByRound	TRUE

Depot Capacity Time Window Rule

Determines which DepotTW an Activity's load is assigned to. Legacy:

earliest DepotTW that overlaps the Shift in AB;

Arrival DepoTW in Scheduling;

EarliestOverlappingShift: earliest DepotTW that overlaps the Shift;

EarliestOfDay: earliest DepotTW that starts within the Day;

Parameter Name	Default
DepotCapacityTWRule	Legacy

Calculate Depot Late Cost by Arrival Time

Set to true to calculate the cost of being late to a depot starting from the arrival time.

Parameter Name	Default
depotLateByArrival	TRUE

First Task Distance Factor

Multiplier of the distance cost applied to the first task on a shift. Promotes the initial assignment of activities close to a crew's starting location.

Parameter Name	Default
firstJobDistanceFactor	1

Optimization Level

Various operations (such as allocating an activity to a shift, initial assignment of a new activity to a shift, etc.) require the sequencing of tasks on a shift to be optimized. This parameter determines how much effort is expended on this kind of optimization. Options are None, Standard, Thorough.

Parameter Name	Default
idbOptLevel	Standard

Immediate Assignment

If checked (True), all mandatory tasks will be assigned to a shift immediately on receipt by the scheduler.

If unchecked (False), tasks will be placed on the free list for later, gradual insertion.

Parameter Name	Default
immediateAssignment	FALSE

Load Factor

Sets the maximum size of goods for a single Activity relative to the vehicle. This is used in the Stop-Shift compatibility check.

Parameter Name	Default
loadFactor	1

Maximum Overlap Between Visits

Maximum allowed overlap between shifts scheduled for the same complex activity, in minutes.

Parameter Name	Default
maxShiftOverlapForMST	15

Maximum Upload Interval

Maximum period (in seconds) between improvement to a solution and its uploading to the database. This is used to ensure that the scheduler periodically saves the improvements to the

current solution to the database so they are available to other processes. If set to zero, every solution improvement is saved to the database immediately.

Parameter Name	Default
maxUpLoadInterval	30

Minimum Travel Time

Minimum travel time (in seconds) between tasks. The scheduler will adjust the travel time if it falls below this value. The system issues a warning if you try to set this value to greater than 600 seconds (10 minutes), as it can have a severely adverse effect on the schedule.

Parameter Name	Default
minTravelTime	0

Number of Tasks Inserted

When there are multiple jobs on the free list, this is the maximum number that will be included in the plan at any one time when not in an Immediate Assignment mode.

Parameter Name	Default
nJobsInsert	10

Over-Scheduling Warning Threshold (hours)

Establishes the threshold, above which an error is generated for overscheduling a complex activity by manually allocating visits..

Parameter Name	Default
overschedThreshold	4

Plan Max Merge Distance

Sets a value to define a distance so that the plan-optimizer creates a new run when the crew travels above that duration. Entered in seconds. This is used to control Scheduler behavior in terms of breaking runs in two.

Parameter Name	Default
planMaxMergeDistance	3600

Plan New Run Probability

Sets a value to define the minimum probability that the plan-optimizer will create a new run. This is used to control scheduler behavior in terms of breaking runs in two.

Parameter Name	Default
planNewRunProbability	0.1

Relative Overcapacity

Depot-window capacity calculated based on relative overcapacity.

Parameter Name	Default
relOverCap	FALSE

Absolute Round Extension

Determines the absolute rounding extension for variable shifts.

Parameter Name	Default
roundExtensionAbs	20.0

Relative Round Extension

Determines the relative rounding extension for variable shifts.f

Parameter Name	Default
roundExtensionRel	.2

Same Site Factor

Multiplier to control allowed ratio between number of sites and number of tasks in a cluster. Do not change without consulting Oracle.

Parameter Name	Default
sameSiteRestrict	5

Shift Window Extension

Number of minutes the scheduler can artificially extend the shift time window for the purpose of optimization when considering compatibility with the tasks. This defines the mismatch allowed between the shift window and task time windows.

If this is set to zero, the shift and task time windows must overlap for allocation to be possible.

If this is greater than zero, then a time window mismatch within this number of minutes will be acceptable to the scheduler when making recommendations.

Parameter Name	Default
ShiftWinExtension	60

Task Late By Arrival

If checked (True), the Late Cost is applied to tasks based on comparison with arrival time.

If unchecked (False), the late cost is applied to tasks based on the predicted departure time.

Parameter Name	Default
stopLateByArrival	TRUE

Time Windows Segmentation

Defines a boundary (in seconds) to segment time between time windows for calculating the time of execution, late/idle time and cost. A value of -1 splits time in the middle between time windows.

Parameter Name	Default
timeWindowsSegmentation	-1

Unexpected Event Handling Threshold

Cost threshold that must be reached in order to activate the Unexpected Event Handling functionality.

Parameter Name	Default
UEC_costThreshold	1.00E+100

Under-Scheduling Warning Threshold (hours)

The threshold, above which a warning is generated for when a complex activity is under-scheduled.

Parameter Name	Default
underschedThreshold	8 hours

Wait Horizon

The time interval from the current time when the Wait Time Cost is applied.

Parameter Name	Default
waitHorizon	2 hours

Optimizer Performance Parameters

Optimizer performance parameters affect how efficiently and effectively the system performs with respect to the scheduler.

Allocation Initial Frequency

The frequency at which the allocation shuffler is activated when the scheduler starts. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
allocMove_initRelFreq	0.1

Allocation Final Frequency

Frequency at which the allocation shuffler is activated when the scheduler finishes. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
allocMove_lastRelFreq	0.1

Allocation Swap Initial Frequency

The frequency at which the allocation swap shuffler is activated when the scheduler starts. This shuffler combines the Allocation and DeAllocation shuffler operations, e.g. exchanges "free" Activities. This is required for visits. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
allocSwap_initRelFreq	0.1

Allocation Swap Final Frequency

The frequency at which the allocation swap shuffler is activated when the scheduler finishes. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
allocSwap_lastRelFreq	0.1

Chain Insert Initial Frequency

Frequency at which the chainInsert shuffler is activated when the scheduler starts. This shuffler tries to assign an entire, or sections of a chain. The default is set to 1.0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
chainInsert_initRelFreq	1.0

Chain Insert Final Frequency

Frequency at which the chainInsert shuffler is activated when the scheduler finishes. The default is set to 1.0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
chainInsert_lastRelFreq	1.0

Chain Move Initial Frequency

Frequency at which the chainMove shuffler is activated when the scheduler starts. This shuffler tries to re-assign an entire chain, or sections of a chain. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
chainMove_initRelFreq	0.1

Chain Move Final Frequency

Frequency at which the chainMove shuffler is activated when the scheduler finishes. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
chainMove_lastRelFreq	0.1

Cluster Initial Frequency

Frequency at which the cluster shuffler is activated when the scheduler starts. The default is set to 1.0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
clusterMove_initRelFreq	1

Cluster Final Frequency

Frequency at which the cluster shuffler is activated when the scheduler finishes. The default is set to 1.0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
clusterMove_lastRelFreq	1

Cost Limit

The scheduler is suspended when the solution cost goes below this value.

Parameter Name	Default
costLimit	0

Deallocation Initial Frequency

Frequency at which the deallocation shuffler is activated when the scheduler starts. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
deAllocMove_initRelFreq	0.1

Deallocation Final Frequency

Frequency at which the deallocation shuffler is activated when the scheduler finishes. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
deAllocMove_lastRelFreq	0.1

Depot Move Initial Frequency

Frequency at which the depotMove shuffler is activated when the scheduler starts. The default is set to 0.5 internally and should not be changed without consulting Oracle.

Parameter Name	Default
depotMove_initRelFreq	0.5

Depot Move Final Frequency

Frequency at which the depotMove shuffler is activated when the scheduler finishes. The default is set to 0.5 internally and should not be changed without consulting Oracle.

Parameter Name	Default
depotMove_lastRelFreq	0.5

Fill Idle Time Initial Frequency

Frequency at which the fill idle shift shuffler is activated when the scheduler starts. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
fillIdle_initRelFreq	0.1

Fill Idle Time Final Frequency

Frequency at which the fill idle shift shuffler is activated when the scheduler finishes. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
fillIdle_lastRelFreq	0.1

Fill Shift Initial Frequency

Frequency at which the fill shift shuffler is activated when the scheduler starts. The default is set to 0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
fillShift_initRelFreq	0

Fill Shift Final Frequency

Frequency at which the fill shift shuffler is activated when the scheduler finishes. The default is set to 0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
fillShift_lastRelFreq	0

Fill Gap Initial Frequency

Frequency at which the fill gap shuffler is activated when the scheduler starts. The default is set to 0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
fillTheGap_initRelFreq	0

Fill Gap Final Frequency

Frequency at which the fill gap shuffler is activated when the scheduler finishes. The default is set to 0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
fillTheGap_lastRelFreq	0

Maximum Exponential Rate

Places a cap on the rise of exponential costs.

When the B value in exponential cost has been used this many times to double the A value, the latter will no longer be doubled.

The default value of 0 denotes true exponential costs.

Parameter Name	Default
maxTerm	3

mstRsrcMove Initial Frequency

Frequency at which the mstRsrcMove shuffler is activated when the scheduler starts. This shuffler tries "single crew" Complex Activities on different Crews. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
mstRsrcMove_initRelFreq	0.1

mstRsrcMove Final Frequency

Frequency at which the mstRsrcMove shuffler is activated when the scheduler's improvement rate slows down. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
mstRsrcMove_lastRelFreq	0.1

Number of Optimization Runs

Number of distinct optimization runs that the scheduler should perform to get itself out of local minima. This applies only when Scheduler Mode is set to either Fixed Real Time or Fixed CPU Time. The default is set to 1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
nOptRuns	1

Cycle Cascade Chance

Average number of additional scheduler subcycles. Setting this to a larger number allows the scheduler to use a broader strategy when searching for cost improvements. If a particular solution appears stagnant, then some additional scheduler sub-cycles may free the restriction. This is set to 0.2 internally and should not be changed without consulting Oracle.

Parameter Name	Default
pcCascadeChance	0.2

Relative Cluster Size

Multiplier for the number of tasks in cluster. The scheduler generally tries to improve the schedule by selecting a set of geographically close activities. This set is referred to as "cluster".

Parameter Name	Default
relClusterSize	1

Run Move Initial Relative Frequency

Sets a value to define the initial relative frequency for the runMove shuffler.

Parameter Name	Default
runMove_initRelFreq	0.1

Run Move Last Relative Frequency

Sets a value to define the final relative frequency for the runMove shuffler.

Parameter Name	Default
runMove_lastRelFreq	0.1

Scheduler Duration

Number of seconds that scheduler should continue to optimize.

If the Scheduler Mode parameter is set to Variable or Forever, this value controls the temperature.

If the scheduler mode is set to Fixed Real Time or Fixed CPU Time, this value limits the optimization time to the required amount of CPU-time or wall-time. Optimization time is also limited by recalculating the temperature based on time left.

Parameter Name	Default
schedDuration	600

Scheduler Heat Rate

Temperature increase/unit drop in cost. The resulting value can be seen in the scheduler log as "t=nnnn." If the resulting value does not rise on making large cost improvements, then scheduler heat rate may need to be increased.

Parameter Name	Default
schedHeatRate	0.5

Scheduler Maximum Temperature

Highest scheduling temperature. Temperature used when the scheduler is started and highest temperature during normal scheduling. Higher temperatures generate better final schedules, but take longer to generate good/acceptable schedules.

Also refer to Scheduler Heat Rate.

Parameter Name	Default
schedMaxTemp	200000

Scheduler Minimum Temperature

Lowest scheduling temperature. Temperature used when scheduling nears completion.

Parameter Name	Default
schedMinTemp	400

Scheduler Mode

The length of time that the scheduler will continue to improve the solution before terminating.

This parameter should be set to FOREVER for seamless scheduling.

- **Forever**: Schedules into infinity, so progress is always 0.
- **Variable**: Finds the remaining amount of time. This is linearly related to the log of the relative temperature.
- **Fixed Real Time**: Scheduling will terminate at the specified wall-time.
- **Fixed CPU Time**: Scheduling will terminate at the specified CPU-time.

Parameter Name	Default
schedMode	Forever

Segment Initial Frequency

Frequency at which the segment shuffler is activated when the scheduler starts. The default is set to 1.0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
segmentMove_initRelFreq	1

Segment Final Frequency

Frequency at which the segment shuffler is activated when the scheduler finishes. The default is set to 1.0 internally and should not be changed without consulting Oracle.

Parameter Name	Default
segmentMove_lastRelFreq	1

Segment Swap Initial Frequency

Frequency at which the segmentSwap shuffler is activated when the scheduler starts. This shuffler exchanges route-segments of two shifts. The default is set to 0.2 internally and should not be changed without consulting Oracle.

Parameter Name	Default
segmentSwap_initRelFreq	0.2

Segment Swap Final Frequency

Frequency at which the segmentSwap shuffler is activated when the scheduler finishes. The default is set to 0.2 internally and should not be changed without consulting Oracle.

Parameter Name	Default
segmentSwap_lastRelFreq	0.2

Shift Initial Frequency

Frequency at which the shift shuffler is activated when the scheduler starts. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
shiftMove_initRelFreq	0.1

Shift Final Frequency

Frequency at which the shift shuffler is activated when the scheduler finishes. The default is set to 0.1 internally and should not be changed without consulting Oracle.

Parameter Name	Default
shiftMove_lastRelFreq	0.1

Tasks In Search Area

Number of tasks that will determine the search area. The search area limits the maximum number of compatible tasks that can be moved by a cluster at one time. The default is set to 50 internally and should not be changed without consulting Oracle.

Parameter Name	Default
stopsInSearchArea	50

Task Grouping Travel Time

Maximum travel radius (in seconds) for a group of tasks. Used for grouping tasks, this determines how far apart tasks can be and still belong to the same group. If the tasks have overlapping time windows and lie within the specified travel radius, then the scheduler will group them for allocation to the same crew.

Parameter Name	Default
tspClassTime	0

Real Time Parameters

Real-time parameters control how the scheduler functions in real-time.

Auto Dispatch Maximum Load

Maximum load as a ratio of vehicle capacity for Auto Dispatch.

Parameter Name	Default
AD_MaxLoad	1.0

Auto Dispatch Minimum Load

Minimum load as a ratio of vehicle capacity for Auto Dispatch.

Parameter Name	Default
AD_MinLoad	0.0

Auto Dispatch Rule

In case goods are scheduled to be picked up at a Depot before the previous run has been completed, this flag controls whether the Scheduler dispatched both runs at the same time, or one by one. ONE_AT_A_TIME: single run only; COMBINED: logical run;

Parameter Name	Default
AD_Rule	ONE_AT_A_TIME

Auto Dispatch Stability Period

Minimum period of stability (in seconds) at a task location before auto dispatching. A stability period is applied to each task that has been nominated for auto dispatching. A task is considered stable if its shift or work sequence has not been changed for at least the specified stability period.

Parameter Name	Default
AD_StabilityPeriod	60

Auto Dispatch Interval

Determines how often (in seconds) the automatic dispatching module checks for tasks to dispatch.

Parameter Name	Default
autoDirectInterval	60

Auto Dispatch On Completion

If checked (True), the scheduler will auto dispatch the next task when the previous task is completed.

If unchecked (False), auto dispatch will occur when the previous task is started.

Parameter Name	Default
autoDirectOnComplete	FALSE

Auto Dispatch Time Horizon

Maximum allowable idle time (in minutes) before automatically dispatching a task. This applies only when Enable Auto Dispatch is True. If the task to be dispatched next will be arrived at more than this number of minutes before it can be started, then it is auto dispatched only after the time period specified here.

Parameter Name	Default
autoDirectTimeHorizon	0

Break Dispatch Mode

Defines the timing in which breaks are automatically dispatched.

Dispatch all at Shift start: All breaks on the started shift are auto dispatched regardless of the shift's drip horizon.

Dispatch all within horizon: Only breaks within the shift's drip horizon are auto dispatched.

Parameter Name	Default
breakDespatchMode	HORIZON

Auto Dispatch Close Distribution Run

Close run when AutoDirect DIST stop.

TRUE stops additional work being added to the run.

FALSE allows additional work to be added to the run.

Parameter Name	Default
closeDistributionRun	TRUE

Emergency Dispatch Mode

Defines emergency task dispatch considerations.

Allow Onsite Crews: An emergency task can be dispatched to a crew even if the crew is working onsite on another task.

Disallow Onsite Crews: An emergency task will not be dispatched to a crew that is currently onsite on another task.

Parameter Name	Default
emergency_despatch_mode	CONSIDER_ONSITE_

Emergency Dispatch Time Left

The scheduler will not consider a crew for an emergency task if the estimated time to complete its current activity is greater than this value. If set to 0, there is no limit.

Parameter Name	Default
emergency_despatch_time	0

Enable Auto Dispatch

If checked (True), auto dispatching of activities is enabled.

If unchecked (False), auto dispatch is disabled.

Parameter Name	Default
enableAutoDirect	TRUE

Non-Productive Task Dispatch Mode

Defines the timing in which non-productive tasks (NPT) are automatically dispatched. Options are:

Dispatch all on shift start: All NPTs on the started shift must be auto dispatched regardless of the shift's drip horizon.

Dispatch all within horizon: NPTs are auto dispatched only if they are within the shift's drip horizon. This is the default.

Parameter Name	Default
pouDespatchMode	HORIZON

POU Dispatch Mode

Defines the timing in which periods of unavailability (POU) are automatically dispatched. Valid modes:

Dispatch all on shift start: All POUs on the started shift must be auto dispatched regardless of the shift's drip horizon.

Dispatch all within horizon: POUs are auto dispatched only if they are within the shift's drip horizon. This is the default.

Parameter Name	Default
meetingDespatchMode	HORIZON

Number of Tasks to be Dispatched

Indicates the number of tasks to be dispatched in horizon-dispatch at a time.

Parameter Name	Default
nMaxDispatchAtATime	1

Process All at Site

If checked (True), all consecutive tasks at a site will be processed all at once.

If unchecked (False), each arrival for the same site task will be processed one at a time. Applies to dead-reckoning only.

Parameter Name	Default
processAllAtSit	TRUE

Real Time Mode

Controls real time and dead-reckoning behavior.

Schedule crews towards their destination: No work can be planned before now and crews are dead-reckoned towards dispatched tasks. This is the default.

Complete activities as scheduled: Same as above, but additionally simulates crews working tasks. Tasks are automatically started and completed at the scheduled times.

Real time off: No simulated crew activity. Crew times and positions remain unchanged.

Parameter Name	Default
realTime	AUTO_POSITION

Crew Location Time Interval

Time interval, in minutes, for relocating the crew by the dead-reckoning subsystem. The location of the crew is determined (in a real-time operation) by dead-reckoning and the occurrence of task and shift milestone events. This parameter also controls the frequency at which a task's schedule is updated for estimated time changes.

Parameter Name	Default
timeInterval	2

Crew Back In Service

Controls when an out-of-service crew is assumed to be back in service.

If set to zero, the crew is automatically assumed to be back in service at the estimated time provided by the crew before going out of service.

If set to a non-zero value, the crew is assumed to be out of service until it actively reports back that it is in service.

Parameter Name	Default
vehicleTimeToActive	0

Wait for Enroute

If checked (True), the system only dead-reckons towards tasks once they become Enroute.

If unchecked (False), the system also dead-reckons towards tasks that are in Closed or Dispatched state.

Parameter Name	Default
waitForEnroute	TRUE

Reference Time Parameters

Reference time is the time used by the scheduler as input to cost-calculations that are sensitive to the current time. These parameters control how and when reference time is updated.

Interval Reference Time

Interval (in hours) with which the reference time is updated. This is set internally to 2 hours and should not be changed without consulting Oracle.

Parameter Name	Default
intervalRT	2

Start Time Reference Time

The time of day at which reference time is synchronized with wall-time. Reference time is used as input to various wall-time dependent functions, like TDC. Reference time is advanced at the interval specified in the Interval Reference Time parameter.

Parameter Name	Default
startTimeRT	0

Scheduler Manager Parameters

Scheduler manager parameters are used to establish various base costs, time values, and other dependencies that affect how the scheduler manages your resources.

Appointment Booking Maximum Cost

Sets a maximum so that the scheduler does not return an Appointment Booking when the cost exceeds this value.

Parameter Name	Default
appointmentBookingCost	50000

Crew Shift Chooser Maximum Cost

Sets a maximum so that the scheduler does not return a shift when the cost of allocating the activity to the shift exceeds this value.

Parameter Name	Default
chooserMaximumCost	50000

Schedule Manager Cleanse Interval

Sets a value, in seconds, to define the interval period at which the Schedule Manager checks to remove old data from the scheduler.

Parameter Name	Default
cleanseInterval	300

Conditional Booking Maximum Cost

Sets a maximum so that the scheduler does not return a Conditional Booking when the cost exceeds this value.

Parameter Name	Default
conditionalBookingCost	50000

Force Use of Site Functionality

Sets a maximum so that the scheduler does not return a Conditional Booking when the cost exceeds this value.

Parameter Name	Default
forceSiteId	FALSE

By default, only activities with a site delay are sent to the scheduler with an identifier attached. When setting the foreceSiteId parameter to TRUE all activities are sent to the scheduler with a site ID defined. The site ID is made up of a concatenation of the address fields of an activity. All activities which have the same address must also have the same longitude/latitude values. For activities to be treated as belonging to the same site they must all have the exactly the same address defined.

Shift Dependency Offset

Sets a value to specify the duration between shifts that causes them to be considered as linked. When this parameter is set to -1 shifts are never considered linked, even if their time windows overlap.

Parameter Name	Default
shiftDependOffset	3600

Site Parameters

Site parameters define the default coordinated universal time and the error translation file used to translate error messages at initialization.

Error Translation File

Directory path and filename of the error translation file used to translate error message text at initialization.

Parameter Name	Default
errorTranslationFile	0

Default UTC Offset

Default UTC offset for packet field, in the format: (-)HHMM or (-hhmmss. The default is the application time zone. The scheduler only deals with absolute, not abstract, times. Use this to pass an optional UTC offset when exporting time data.

Parameter Name	Default
hipDefaultUTCoffset	SystemTZ

Index

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W