



The information in this document is subject to change without notice and should not be construed as a commitment by Ascari Limited. While all reasonable care has been taken to ensure accuracy, Ascari Limited assumes no responsibility for errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

Copyright © 2006 Ascari Limited. All rights reserved.
First Edition (2006) version 1.0

No part of this document may be reproduced in any form or by any means without written permission from Ascari Limited.

Trademarks used in this document are the property of their respective owners.

Table of Contents

Introduction to the Equipment Manager	5
What is the Equipment Manager?	5
Equipment Manager Overview	6
The Connection Manager	7
The Scenario Manager	7
The Equipment Event Manager	7
The Application Event Manager	7
The Equipment Status Manager	8
The Alarm Manager	8
Configuration of Each Component	8
Equipment Manager Operation	10
Connection Manager	10
Scenario Manager	11
Event Manager	12
Application Event Manager	12
Equipment Status Manager	12
Alarm Manager	13
Data Dictionary	13
The SECS-II Library	15
Configuring the Equipment Manager.....	17
Custom XML Driver File	17
Configure Communications	17
Configuring HSMS Communication	17
Configuring SECS-1 Communication	17
Configuring TCP-IP/TELNET Communication	18
Processors of the Equipment Manager	18
Establish Communications	18
Setup Constants	19
Define Reports	19
Supported Scenarios	20
Scenario Operations	21
Operation Mappings	22
Application Events	23
Equipment Status Monitor	23
Event Manager	23
Alarm Manager	24
Appendix – Example ‘Equipment.properties’ File	25
Appendix – Example ‘XML Library’ File.....	30

Who Is This Guide For?

This guide is intended for the person who performs all or some of the following functions:

- Equipment Characterisation
- Equipment Automation Development
- Equipment Automation Implementation & Support

This guide will refer to the following related manuals:

Additional Copies of This Guide

If you want additional printed copies of this Guide please contact Ascari at www.ascari-it.com

Introduction to the Equipment Manager

What is the Equipment Manager?

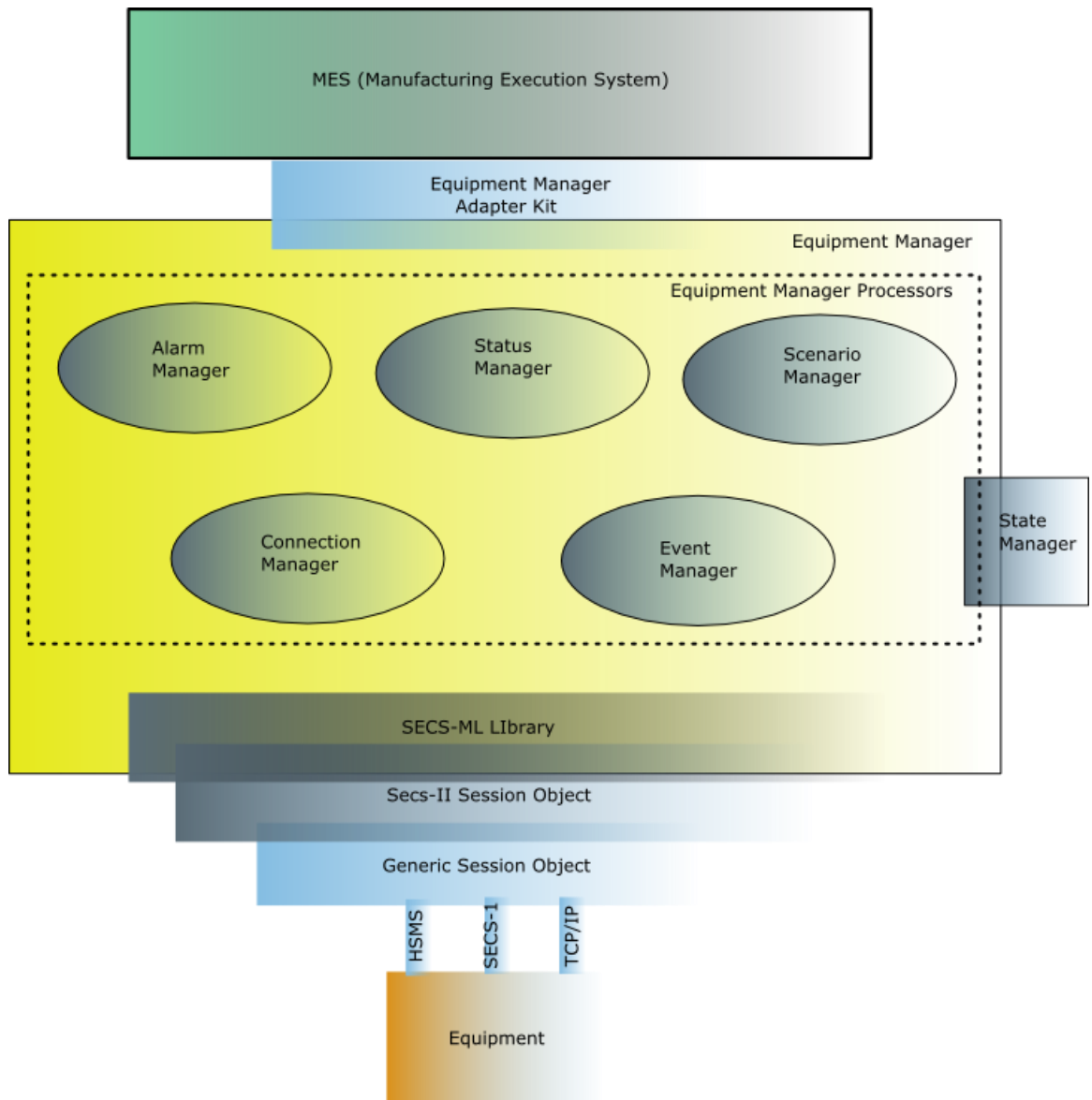
The Ascari Equipment Manager is a JAVA™ application that allows rapid implementation of automated equipment interfaces. The Equipment Manager provides a number of standard 'out of the box' configurable components:

- Standard GEM compliant process for handling Equipment Communication
- Standard GEM compliant process for handling Equipment Status monitoring
- Standard GEM compliant process for handling Equipment Event handling
- Standard GEM compliant process for handling Equipment Alarm handling
- Standard GEM compliant process for executing an Operational Cycle

By configuring each of these standard components, the Ascari Equipment Manager provides a generic configurable framework for automating SECS-II compliant equipment and allows equipment specific automated scenarios to be implemented without the need for custom development.

The Ascari Equipment Manager has been developed to allow any of the standard components and functionality to be extended, using the supplied API, in order to provide additional equipment specific functionality not covered by the framework.

Equipment Manager Overview



The Ascari Equipment Manager comprises of a number of pre-built and configurable processes which provide key standard functionality for implementing an automated equipment scenario. The following sections give an overview of each of these standard components (or managers).

The Connection Manager

Each equipments SECS-II interface will likely have differing connection properties for establishing itself upon a network and providing a state in which remote communication is possible. The connection manager supports the three standard modes of communication to a SECS-II interface:

- SECS-1 via RS-232.
- HSMS
- TCP-IP/TELNET

Each mode is configurable within the Equipment Manager depending on the automation requirements of the target equipment. The Equipment Connection Manager component is responsible for the creation and management of a SECS-II connection that allows communication between the Equipment Manager and the target equipment via one of the above modes of communication.

The Scenario Manager

For the automation of any SECS-II equipment the operational scenario for the equipment first requires definition. An 'equipment specific' scenario can be thought of as the entire operational cycle from start to end, 'Locking', 'Loading', 'Selecting', 'Starting', 'Processing', 'Unloading' and 'Unlocking' to allow the complete processing of product. This scenario can be split into multiple "operations" to provide control over the process via monitoring the status of an equipment such as equipment actions, events, status variables etc.

As an example, a complete operational scenario could be split into two sub scenarios 'LOAD' and 'UNLOAD', each incorporates a number of operations. Thinking in terms of a 'LOAD' cycle and an 'UNLOAD' Cycle, 'Locking', 'Loading', 'Selecting' and 'Starting' will be operations of the 'LOAD' cycle and 'Unloading' and 'Unlocking' will be operations of the 'UNLOAD' cycle. Depending on the equipment type/characteristics and the software revision it may be the case that there are many operations within each cycle or scenario (as above) or alternatively it may be the case that only one operation is required in order to begin the processing off, or unloading off, WIP.

It is the role of the Scenario Manager component to execute equipment specific scenarios using an embedded state machine to control the automated operation of the target equipment.

The Equipment Event Manager

An equipment specific scenario will entail a number of 'Collection Events' received from the equipment during its operational scenario. It is these events that are required in order to trigger 'state changes' within the Equipment Manager Scenario Manager.

It is the role of the Event Manager component to listen to these Collection Events and, if the event is of interest, to fire an Internal Event in order to trigger some action e.g. 'State Change' within the Equipment Manager.

The Application Event Manager

An Application Event is a message that is sent to the host application, based upon the outcome of an Operation or Reception of a particular Equipment Collection Event. Application Events are used for actions such as:

- Informing the MES when an action has completed on the Equipment or within the Equipment Manager

- Notifying any host applications when an alarm occurs on the equipment.
- Data Collection etc.

The Application Event Manager component is responsible for firing Application Events from the Equipment Manager to any registered “listeners”.

The Equipment Status Manager

Each equipment will have a number of equipment specific ‘Status Variables’. These status variables change their values during the operational scenario of the equipment and can be used to determine if we have ‘Good to Go’ values such that a specific scenario or operation can be performed. Status variables may reflect the equipments ‘Process State’ or ‘Control State’ and can be used to determine if the equipment is in the correct state to proceed. It is the role of the Status Manager component to monitor and update any Status Variables that are of interest such that the Equipment Manager can keep track of the current state of the target equipment.

The Alarm Manager

As the majority of process equipment are extremely complex, in terms of their operation, there are a vast number of exceptions that can occur during processing. These exceptions may require some form of human intervention at the equipment before the processing of WIP can continue.

When an equipment exception occurs, the equipment usually notifies about the exception by raising an Alarm Report via the SECS-II connection, usually combined with an audible tone and/or a flashing red light on the signal tower.

It is the function of the Alarm Manager to listen for Alarm Reports output from the equipment such that the Equipment Manager can handle the exception and notify any integrated applications.

Configuration of Each Component

Each of the above components of the Equipment Manager requires some level of configuration such that they can function correctly for the specific equipment the equipment manager is used to control. For example, the Connection Manager component requires configuration in terms of its communication mode and respective parameters.

In order to achieve this the Equipment Manager makes use of a properties file. The properties file is made up of a number of sections each corresponding to a particular component of the Equipment manager. The “

Configuring the Equipment Manager” section details how the equipment manager can be configured using the properties file to implement an equipment specific automated operational scenarios. The Appendix also provides an example of the properties file.

Equipment Manager Operation

Following the general overview of the equipment manager components, the following section covers in further detail the individual operation of the standard equipment manager processes.

Connection Manager

The Equipment Connection Manager provides a ‘Generic’ implementation of the GEM Equipment Connection State. It is the Equipment Connection Manager that is responsible for utilising a number of connection parameters in order to establish a connection, to an equipment, at the SECS-II level. The Equipment Connection Manager will also maintain the connection status of the equipment’s SECS-II interface to the equipment manager.

Upon activation the Equipment Connection Manager will determine the current connection status between the equipment manager and the equipment’s SECS-II interface and if required establish a low level communications link to the equipment.

Once a successful connection is established the Equipment Connection Manager, depending on the ‘Connection Mode (Active/Passive), may perform a connection to the SECS-II interface of the equipment using the SECS-II Transaction specified in the properties file.

If the Connection Mode for the Connection Manager is set to ‘Passive’ then the Connection Manager will attempt the connection and set the connection status to ‘Communicating’ upon the receipt of a response from the target equipment. The Equipment Connection Manager will then start a ‘Heartbeat Timer’ such that the connection can be monitored and maintained. The Equipment Connection Manager, once connected and communicating, listens to internal “Session Events”. A Session Events is raised by the low level SECS-I protocol driver used within the equipment manager to notify of a ‘Connect’ or ‘Disconnect’ between the equipment manager and the SECS-II interface. When a Session Event is raised the Connection Manager sets its Connection State to ‘Enabled’ or ‘Disabled’ appropriately.

If the Connection Mode for the Connection Manager is ‘Active’ then the Connection Manager waits for the equipment to establish communications. The Equipment Connection Manager assumes the status ‘Communicating’ and then waits for the equipment to connect. SECS-II Message Events from the Equipment are monitored within the Equipment Connection manager such that, in the event of an ‘Active’ connection, the Manager can receive the handshake from the equipment and initiate the ‘Heartbeat Timer’ such that the connection can be monitored and maintained.

The Equipment Connection Manager is also responsible for monitoring any timeouts defined for the SECS-II connection session maintained and the target equipment. If a timeout is raised then the Equipment Connection Manager will take the appropriate action in compliance with any timeout configuration parameters and the SECS standard.

Once the Connection Manager is connected to a specific equipment the Equipment Connection Manager will run through an initialisation cycle that will perform a number of initialisation tasks upon the target equipment such that it is ready for the automated processing of WIP. The standard sequence of tasks performed is:

- Set the date & time on the equipment, if required
- Set the value of any equipment constants on the equipment which are required within the equipment manager
- Disable all events on the equipment (The Event Manager will enable the specific events required for the equipment manager)
- Delete any existing defined reports on the equipment if required
- Create any reports
- Link reports on the equipment to defined collection events

However, the list of actions performed on initialisation of the equipment manager can be configured from within the properties file.

The final function of the Equipment Connection Manager is to create the 'Data Dictionary' (discussed later) used for maintaining the status of any Status Variables of interest and any associated Collection Events.

Scenario Manager

It is the function of the Scenario Manager to execute and control Scenarios that have been specified within the equipment specific properties file depending. Each scenario defined for the Equipment Manager, may comprise and accept a number of 'Scenario Parameters' which can be passed from an integrated application such as an MES.

Scenario Parameters can be defined as any parameters required by the Equipment Manager, in order to successfully execute one or more Operations within a given Scenario.

For example the Scenario 'LOAD' may have a number of Operations defined, one of which is 'SELECT_RECIPE'. This 'SELECT_RECIPE' operation, once executed, will result in a transaction being sent to the target equipment such that the process program can be selected. In this example we will assume that the transaction requires two parameters in order to achieve recipe selection, these parameters are the 'MID' or 'Material ID' and 'PPID' or 'Process Program ID'. It is these parameters that will be passed to the Scenario Manager as 'Scenario Parameters' from the MES, for example, such that the above Operation can be correctly executed and the appropriate data sent to the target equipment as part of a specific transaction.

Scenarios are specified within the equipment specific properties file along with their associated Operations and any associated Operation properties. Once the Scenario Manager is created it will access the properties file and read in the list of Scenarios that may be executed for this equipment type. As the Scenario Manager has a notion of any Scenarios that it can execute along with a list of Scenario Parameters, any of the specified and configured Scenarios can be executed from any integrated application such as an MES.

For example focussing on the aforementioned 'LOAD' and 'UNLOAD' Scenarios assume the following:

The 'LOAD' Scenario at the Equipment Manager level will be triggered by a 'LOAD_EQP' action at the MES level and the 'UNLOAD' Scenario will be triggered by a 'RESOURCE_RELEASE' action within the MES.

For each of the MES actions defined above an API call is made to the required equipment manager containing the name of the scenario to execute together within any required scenario parameters. The

Scenario Manager then executes the defined operations for the requested Scenario. The “Configuring Scenario Parameters” section later in this document provides further details on how the Scenario Manager can be used and configured, including waiting for specific tool events to occur on the equipment.

Event Manager

The Equipment Event Manager listens to Collection Events from the target equipment and, providing the Collection Event exists within the configured equipment manager library or XML driver file, fires an Internal Event to notify all subscribing Equipment Manager Components.

The Event Manager manages any Internal Events and keeps track of subscriptions to raised events, events can then be sent upwards to all subscribers. The Event Manager (not an Equipment Manager processor) spawns multiple processes for each Internal Event received such that it can publish each event as a separate thread.

Application Event Manager

An Application event is an Event sent to the host application pending the output of a certain Operation, receipt of a Collection Event, an Alarm raised by the equipment or an error occurring within the Equipment Manager. The Application Event Manager handles the processing of Internal Events such that, should a mapping to an Application Event exist, an Application event can be raised and sent to all external applications. The external applications will have subscribed to these application events for the associated Equipment Manager.

An Application Event is automatically raised when a configured Alarm is raised on the equipment or an error occurs within the Equipment Manager. However, for Application Events to be raised when a specific Equipment Collection Event is received by the Equipment Manager or a Scenario Operation is completed individual configuration of the Application Event names and mappings required definition. The configuration section contains specific details.

Equipment Status Manager

In simple terms the function of the Equipment Status Manager is to monitor and manage the current status of the target equipment. The Equipment Status Manager loads in all Status Variables specified in the equipment specific properties file and will add them, plus their associated Collection Events, to the Data Dictionary object if they do not already exist.

On initialisation of the Equipment Manager, the Equipment Status Manager updates status variables by creating a SECS-II Transaction (usually an S1F3) that specifies all Status Variables (SVIDs etc..) of interest in the Primary Message of the Transaction. It is the Secondary Message that, when returned from the equipment, contains the values for all of the Status Variables specified in the Primary. The Equipment Status Manager can then update the Data Dictionary with the newly returned Status Variable Values.

In addition to the above, the Equipment Status Manager also listens to Internal Events fired internally within the Equipment Manager. If the internal event represents a Tool Collection Event sent by the tool the Status Manager will determine if the Collection Event is “mapped” to a monitored status variable. If the event is mapped then the Equipment Status Manager obtains the latest value(s) of the Status Variable(s) from the tool and updates the Data Dictionary with any new Status Variable values.

In basic terms the Equipment Manager performs a number of Operations for specific Scenarios throughout an Operational Cycle. Any number of these operations could trigger a State Change within the target equipment that may result in a change in value of any one of the Status Variables in which we are interested. Certain Operations can be dependant upon Status Variables holding specific 'Good to Go' values and thus it is of vital importance that the Data Dictionary be updated whenever it is likely that Status Variable values may have changed.

It is for the above reason that after a specific Operation, if an update of Status Variables is required, an Internal Event will be fired into the Equipment Status Manager. Upon receipt of this Internal Event the equipment Status Manager can determine if the event is of interest and should cause a Status Variable update and if so will update the Data Dictionary by requesting a Status Variable update from the target equipment in the way described above.

Alarm Manager

The Alarm Manager component of the Equipment Manager functions as a listener and will process any alarms output by the target equipment during an Automated Operational Cycle. The Alarm Manager uses its configuration properties to determine any actions that should be taken, if any, on receipt of an alarm.

If an alarm occurs that should cause the Equipment Manager to abort its current Operation or Scenario the Alarm Manager will fire an Internal Event to that effect. Any Equipment Manager components with an Event Listener will pick up the Internal Event and, upon determining that the event in an Alarm Report, take the appropriate steps such that the Operation or Scenario can abort safely.

Data Dictionary

The Data Dictionary can be thought of as an Object within the Equipment Manager that provides access to some of the components specified within the equipment specific properties file. For example the Data Dictionary will contain sections for both Status Variables and Collection Events storing information such as the SVID or CEID and their up to date values in sync with the current status of the target equipment.

Data Sections and Data Items can be added to the Data Dictionary as and when required via its various methods. This is achievable as it is possible for the Data Dictionary to create new instances of both Data Sections and Data Items providing a means to customise and extend the contents of the Dictionary. This may be desirable when extending the functionality of the Equipment Manager

As the Data Dictionary holds information about the current status of the target equipment, Status Variables can be queried directly from the Data Dictionary when checking 'Good to Go' values prior to the execution of a specific Operation. These Status Variable values are updated within the Data Dictionary upon the receipt of certain Collection Events by sending a SECS-II transaction, defined within the tool XML library, to the equipment.

In order to build, and make reference of, the contents of the Data Dictionary, each Equipment Manager processor makes reference to its specific section depending on the nature of the processors function. Each

The SECS-II Library

When automating an equipment of any type it is the case that the automation component, in this case the Ascari Equipment Manager, is required to communicate with the target equipment via its SECS interface. This SECS communication protocol is governed by the 'SEMI E5' standard or SECS-II Standard (SEMI EQUIPMENT COMMUNICATIONS STANDARD 2 MESSAGE).

It is this standard that provides a specification as to how to communicate with an equipment during an automated cycle and specifies detailed reference messages. With each equipment that is termed to be 'SECS Enabled', a software revision is shipped along with an accompanying SECS manual.

The software revision on the target equipment will make use of a SECS-II interface that is usually configurable at the equipment. It is this SECS-II interface that makes communication possible between an Equipment Manager and the target equipment.

The SECS manual shipped with the equipment provides equipment and software revision specific information based around the SECS-II definitions and message detail that are required to run that particular equipment in an Automated Scenario. There are a number of SECS-II components that are of interest to us when automating an equipment:

- *SECS Elements*
- *Status Variables*
- *Data Variables*
- *Collection Events*
- *Reports*
- *Alarms*
- *Equipment Constants*
- *Transactions*

All of the above are specified in the SECS manual shipped with the software revision of the target equipment along with information on their detail and usage within an Automated Scenario.

In order for the Equipment Manager to execute an Automated Scenario it must have the ability to access each of the above objects from a predefined Library created specifically in accordance with the target equipments software revision and SECS specification.

Each configured Equipment Manager uses a specific XML defined library file within the SECS-ML layer of the equipment manager for communicating with the individual equipment the equipment manager controls. The XML library is created in the characterisation phase of the automation project using the Characterisation GUI provided by Ascari Limited.

The XML Library is structured such that there is a specific Library Section for each of the above SECS-II definitions. Each Library Section is then populated during equipment characterisation with the various SECS Elements, Status/Data Variables, Collection Events, Equipment Constants and Transactions etc, required for the execution of an Automated Operational cycle or Scenario for this specific equipment or software revision. It is this XML library file, and the relevant section within it, which is used by the Equipment Manager

components described above for mapping the generic operation of the Ascari Equipment Manager to the specific configuration required for an individual equipment.

Note. Only the SECS-II definitions required for the automated Scenario are required within the XML library file. Therefore, only a subset of the supported SECS-II definitions of specific tool are likely to be required resulting in a reduced “footprint size” for the XML library. The “Characterisation GUI User Guide” provides further information on creating a tool XML library. The Appendix also provides an example of a defined XML library.

Due to the nature of the Characterisation GUI the requirement should never exist to open an XML file for editing in any other package than the Characterisation GUI. i.e. No XML knowledge is required in the building of the equipment Manager Library or the configuration and operation of the Ascari Equipment Manager.

Configuring the Equipment Manager

Custom XML Driver File

This is the driver for the Equipment Manager and contains all the required SECS-II data and messaging information required to perform an automated scenario for the target equipment. This file is produced by the Characterisation GUI and is required by the Equipment Manager as a reference. The custom XML driver should be specified in the properties file as follows:

Equipment.library = c:|\eqp.xml

Configure Communications

This section of the properties file provides the SECS communication parameters required for connectivity to the target equipment via the desired protocol. This section relates directly to the Connection Manager component of the Equipment Manager. There are three main modes of communication each requiring slightly different configuration parameters. Below are descriptions of the required properties for each of the three modes of communication:

Configuring HSMS Communication

The settings in this menu must correspond to the settings in the associated configuration for the port on the equipment.

Connection Mode	PASSIVE or ACTIVE.
Local Port	The port upon the local device (where the EM is running)
Remote Port	The port upon which the target equipment listens remotely.
Local IP	The IP address of the local device (where the EM is running).
Remote IP	The IP address of the equipment targeted for characterisation.
Link Test Timer	Length of time for lower level heartbeat (zero value means no heartbeat).
HSMST3	Timer T3: Time period for reply timeout in the HSMS protocol.
HSMST5	Timer T5: Time period for connect separation timeout in the HSMS protocol.
HSMST6	Timer T6: Time period for control timeout in the HSMS protocol.
HSMST7	Timer T7: Time period for connection idle timeout in the HSMS protocol.
HSMST8	Time T8: Time period for network inter-character timeout in the HSMS protocol.
Device ID	A 15-bit field used to identify the device.

Configuring SECS-1 Communication

RS232 Equipment connection.

Baud	The baud rate of the port.
Retry Limit	The number of retries.
Port	The port that is to be used for the communication on your own laptop or pc.

<i>SECSIT1</i>	<i>Timer T1: Time period for inter-character timeout in the block transfer protocol.</i>
<i>SECSIT2</i>	<i>Timer T2: Time period for timeout in the block transfer protocol.</i>
<i>SECSIT3</i>	<i>Timer T3: Time period for reply timeout in the message protocol.</i>
<i>SECSIT4</i>	<i>Timer T4: Time period for inter-block timeout in the message protocol.</i>
<i>Device ID</i>	<i>A 15-bit field used to identify the device</i>

Configuring TCP-IP/TELNET Communication

If communication with the equipment is via TCP-IP/TELNET then the following menu option should be used:

<i>Remote Port</i>	<i>The port upon which the target equipment listens remotely.</i>
<i>Remote IP</i>	<i>The IP address of the equipment targeted for characterisation.</i>
<i>Retry Limit</i>	<i>The number of retries.</i>
<i>SECSIT1</i>	<i>Timer T1: Time period for inter-character timeout in the block transfer protocol.</i>
<i>SECSIT2</i>	<i>Timer T2: Time period for timeout in the block transfer protocol.</i>
<i>SECSIT3</i>	<i>Timer T3: Time period for reply timeout in the message protocol.</i>
<i>SECSIT4</i>	<i>Timer T4: Time period for inter-block timeout in the message protocol.</i>
<i>Device ID</i>	<i>A 15-bit field used to identify the device</i>

Processors of the Equipment Manager

The Equipment Manager is a concurrent application and is comprised of a number of individual threads or processors as shown in the diagram at the beginning of this document. This section of the properties file allows the specification of the Equipment Manager processes that should be started for this particular equipment type. Effectively processes can be turned on, or switched off, depending upon the requirement of the scenario. A list of desired processors should be specified in the properties file as follows:

*Equipment.Processors = EquipmentEventManager ConnectionManager
EquipmentStatusManager ApplicationEventManager*

All processes in the above list will be enabled when the Equipment Manager is initialised.

Establish Communications

Depending on the 'Equipment Connection Mode' parameter and the mode of communication, communication with the equipment can be initiated by either the Equipment Manager or by the equipment itself. As the majority of equipments are generic in terms of their SECS connection it will largely be the case that communications will be established between Equipment Manager and equipment via the transaction S1F13 – 'Establish Communications'. However it may be the case that the equipment establishes communications via a different SECS-II transaction (e.g. S1F1) and thus this transaction should be made configurable within the Equipment Managers properties file.

The transaction used to establish communications should be specified by stream and function as follows:

Equipment.establishCommunications = SIF13

Setup Constants

Equipment Constants can be equipment specific and can be used to configure differing characteristics of an equipment for an automated scenario. For example there may be a number of Equipment Constants that restrict the operator from performing certain actions at the equipment's interface when in a 'remote' mode of operation. Another example may be a series of Equipment Constants that hold the string name of a particular component of the equipment that may be referenced by the Equipment Manager during an automated Scenario. This could be, for example, a load port or a tray position upon a carousel. With the latter example it may be the case that, in order to fulfil the requirements of our automated scenario, we need to specify a specific string value to each tray position on the carousel. Setting up any constants on the target equipment should be done by completing this section of the properties file, an example of how this may be achieved is shown below:

Equipment.constants = TRAY_POS_1_NAME

Equipment.constant.TRAY_POS_1_NAME.name = Tray1Name

Equipment.constant.TRAY_POS_1_NAME.value = Tray1

Define Reports

It will be the case that the equipment manager will enable certain Collection Events and, upon receiving them, will likely change in state and/or update one or more Status Variables. It may also however be the case that one or more of the Collection Events has some associated Report Data with it, Report Data that may be required by the Equipment Manager in order to perform a specific task. For example the Collection Event 'Tray Home', output when the tray is returned from the process chamber, may have an associated Report that holds data providing the id of the tray position to which the tray in question has homed. The Equipment Manager may require this tray position such that an 'UNLOAD' scenario can be executed for the acquired tray position. Alternatively a report may contain data which requires passing up to an integrated application such as an MES for storing against a WIP or Equipment unit modelled within the application.

Report Data held in Data Variables (DVIDs) are associated to a Report, which is in turn associated to the collection event upon which the report came in. It is for this reasoning that Reports should be defined in the Equipment Managers properties file as follows:

Equipment.reports = r1 r2 r3

Equipment.report.r1.name = --- the name of the report

Equipment.report.r1.rptid = --- the report id

Equipment.report.r1.event = ev1 ---- the name of the collection event from the driver

Equipment.report.r1.variables = v1 v2 v3 ---- a list of data variables that are contained in the driver

Equipment.report.r2.name = --- the name of the report

Equipment.report.r2.rptid = --- the report id

Equipment.report.r2.event = ev2 ---- the name of the collection event from the driver

Equipment.report.r2.variables = v1 v2 v3 ---- a list of data variables that are contained in the driver

Etc.....

Supported Scenarios

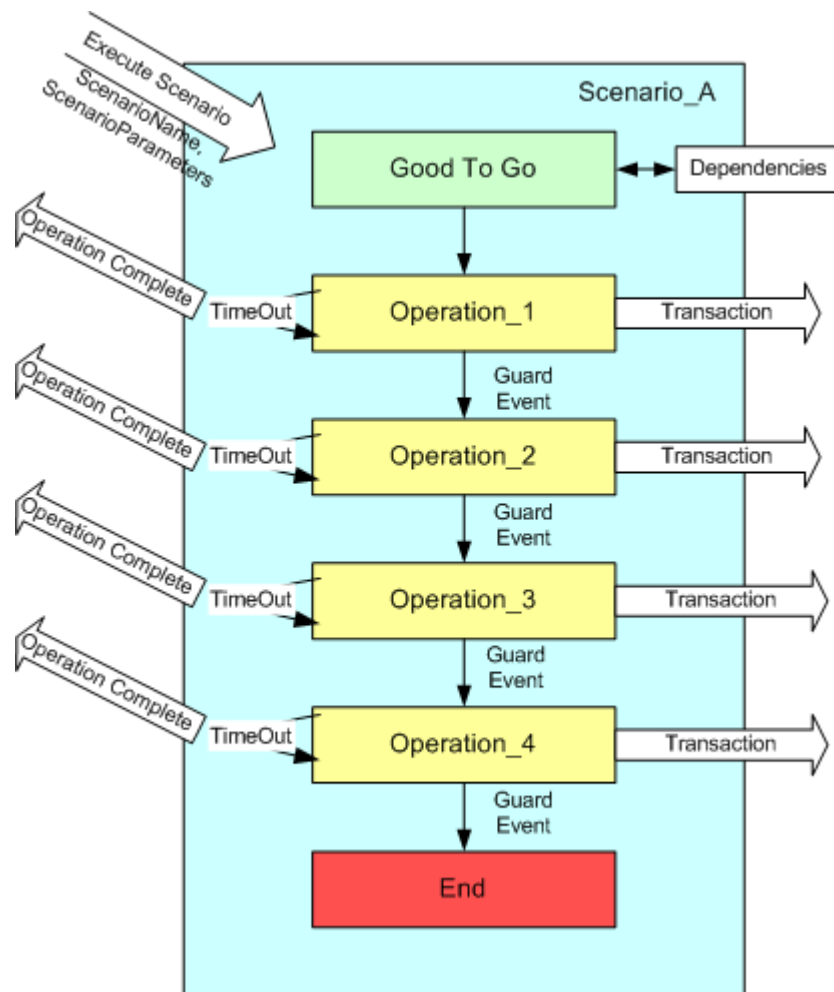
Automated Operation Scenario may comprise of a number of sub Scenarios such as ‘LOAD’ and ‘UNLOAD’. It is this particular section of the properties file that will allow the specification of the Scenarios required to build up our complete Automated Scenario. Example:

Scenarios = LOAD UNLOAD

In the majority of cases a ‘LOAD’ and an ‘UNLOAD’ scenario will be sufficient in order to complete an Automated Cycle however, it may be the case that the target equipment requires some additional logic that is not part of either Scenario. If this is the case then additional Scenarios can be specified in this section of the properties file.

Scenario Operations

Once the required scenarios have been specified it is time to think about the operations associated with those scenarios. Each scenario can be comprised of one or more operations, one at a minimum for less complex equipments, to many operations for more complex equipments that may require a larger number of Remote Commands in order to step through the Automated Cycle.



Scenario Overview

In basic terms a Scenario should be allocated a list of Operations (1 or more). In addition a Scenario may also specify a list of dependencies (or Status Variables) upon which the execution of the Scenario depends. Any defined Status Variables dependencies, are allocated a list of 'Good to Go' values which determine whether or not the Scenario can proceed.

The following example, shows a 'LOAD' Scenario can not be performed until the Status Variable 'TRAY_1_STATUS' has a value equal to one or more of the specified (space delimited) 'Good to Go' values.

Operations are specified in the properties file as follows:

```
Scenario.LOAD.operations = SELECT_RECIPE  
Scenario.LOAD.dependencies = TRAY_1_STATUS  
Scenario.LOAD.goodToGo = 1 7 9
```

In the above case the 'SELECT_RECIPE' Operation of the 'LOAD' Scenario will not be executed until the status of the Status Variable 'TRAY_1_STATUS' has a value of 1, 7 or 9. The values specified could indicate the following:

```
1: Cassette in Tray 1 Ready  
7: Cassette in Tray 1 Idle  
9: Cassette in Tray 1 Ready Select
```

A value other than any of the specified values could indicate that no WIP has been loaded into the target equipment and thus a recipe can not be selected by the Equipment Manager.

Operation Mappings

Once 'Good To Go' values have been obtained, Operations defined for a Scenario are sent to the equipment by the Equipment Manager in the form of a SECS-II Transaction. With reference to the above Operation the SECS-II Transaction required to execute the selecting of a recipe or process program will have been engineered during characterisation and will exist in the XML Driver File. This SECS-II Transaction will have been allocated a String name, 'PP_SELECT' for example.

Once the Operation has been executed on the equipment it is usually the case that the equipment will output a Collection Event to signify that the Operation has been carried out successfully. It is the SECS-II Transaction name and SECS-II Collection Event name together with a time, if required, in which the event should be received, that must be mapped to the Operation within the properties file as follows:

```
Operation.SELECT_RECIPE.transactionName = PP_SELECT  
Operation.SELECT_RECIPE.eventName = RECIPE_SELECTED,600
```

The above example shows the SELECT_RECIPE operation will result in the PP-SELECT transaction defined within the XML library for the equipment will be executed by the Scenario Manager to perform the SELECT_RECIPE operation. The Scenario Manager will then "wait" until the RECIPE_SELECTED event defined within the XML library is received before proceeding to the next Operation defined for the scenario, if any. If the RECIPE_SELECTED event is not received within 600 seconds the SELECT_RECIPE operation will timeout and the operation and Scenario will be aborted by the Scenario Manager.

The Scenario Manager allows an Operation to be defined without a transactionName property. In this case the Operation, will not perform any transaction and will just wait for a specific event to be received before being marked by the Scenario Manager as complete.

Application Events

Application Events are sent to the Host Application pending the outcome of an Operation or upon the receiving of a Collection Event. This section of the properties file can be used to map Application events to both Collection Events defined within the XML library for the tool and Operations executed within the Scenario Manager.

The properties below defines two application Events. An application event “Event1” is sent to any integrated applications when the “Process State Changed” defined within the XML tool library is received from the equipment. The second application event “Event2” is raised when the Scenario Manager has successfully executed the defined “SELECT_RECIPE” operation.

```
ApplicationEvents = Event1 Event2  
ApplicationEvent.Event1.eventName = Process State Changed  
ApplicationEvent.Event1.operationName = SELECT_RECIPE
```

Equipment Status Monitor

The Equipment Status Monitor is responsible for maintaining the correct status of the equipment via the Equipments Status Variables. The Equipment Manager is able to keep track of the current status of all of the Status Variables of interest via specific events output from the equipment and by Status Variable requests made upon the equipment.

This section of the properties file allows us to specify the Status Variables of interest and the Collection Events upon which they will be updated as follows:

```
Status.variables = ControlState ProcessState  
Status.ControlState.event = Control State Changed  
Status.ProcessState.event = Process State Changed
```

This results in the current value of “ControlState” status variable being obtained from the equipment when the “Control State Changed” event defined within the XML tool library is received from the equipment. Likewise, the current value of “ProcessState” status variable is obtained from the equipment when the “Process State Changed” event defined within the XML tool library is received from the equipment.

Event Manager

The Equipment Event Manager, “listens” for any Equipment Events which have been defined within the Equipment Manager as of interest to any of the specific Equipment Manager processes. On receipt of a defined Equipment Collection Event the Event Manager raises an “Internal Event”, containing details of the Equipment Collection Event into the Equipment Manager. The following configurable property defines which SECS-II Transactions received from the Equipment will be monitored for notifying the equipment manager of the defined events of interest.

ToolEventManager.EventMessages = S6F11 S6F13

Alarm Manager

An alarm may occur at any point during processing. Alarms can vary in their severity and importance with regards to the Automated Cycle. Upon configuring the alarms section of the properties file we may wish to do the following:

- Enable all alarms on the equipment
- Provide a list of alarms to ignore
- Enable only specific alarms
- Ignore alarms of a specified severity
- Specify transactions upon which alarms occur

This is done within the properties file as follows:

AlarmManager.enableAll = true/false
AlarmManager.ignoreAlarms = alarm1 alarm2
AlarmManager.enableAlarms = alarm1 alarm2 ...
AlarmManager.ignoreSeverity = severity1 severity2
AlarmManager.AlarmMessages = trans1 trans2

Alarms are enabled and disabled using specific SECS-II transactions that will exist within the XML driver file. When configuring alarms within the properties file the names of these transactions should be specified for each alarm action. The string name specified should be the name of the transaction within the XML driver file and should be specified as follows:

AlarmManager.enableAlarms.transactionName = Enable Alarms
AlarmManager.enableAll.transactionName = Enable All Alarms

Appendix – Example ‘Equipment.properties’ File

Example properties file that will populate Data Dictionary.

```
#
# Copyright (c) 2003 Ascari Limited. All Rights Reserved.
#
# This software is the confidential and proprietary information ("Confidential Information") of
# Ascari Limited. You shall not disclose such Confidential Information and shall use it only in
# accordance with the terms of the license agreement you entered into with Ascari Limited.
#
# Ascari Limited makes no representations or warranties about the suitability
# of the software, either express or implied, including but not limited
# to the implied warranties of merchantability, fitness for a particular
# purpose, or non-infringement. Ascari Limited shall not be liable for any damages
# suffered by licensee as a result of using, modifying or distributing
# this software or its derivatives.

##
# Equipment specific configuration
##

# the custom XML library for use with this instance of the EM
Equipment.library = c:\|eqp.xml

# 0 = SECS-1, 1= HSMS, 2=SECSI over TCP/IP
Equipment.portType = 1
Equipment.localAddress = 127.0.0.1
Equipment.remoteAddress = 127.0.0.1
Equipment.localPort = 5000
Equipment.remotePort = 5001
Equipment.connectionMode = PASSIVE
Equipment.linkTestTimer = 30
Equipment.T3 = 45
Equipment.T5 = 10
Equipment.T6 = 5
Equipment.T7 = 5
Equipment.T8 = 20
Equipment.heartbeat = 0
```

```
# Defines the list of processes the EM should run
Equipment.Processors = EquipmentEventManager ConnectionManager
EquipmentStatusManager ApplicationEventManager

# The following parameter dictates the transaction used to establish comms with the equipment
# it defaults to SIF13
Equipment.establishCommunications = SIF13

##
# Equipment Constants Section - this allows a list of equipment constants to be set up
# on the equipment whenever it goes through its initialisation
#
#Equipment.constants = c1 c2 c3 c4 .....
#Equipment.constant.c1.name = XXXX - name of the constant in the library
#Equipment.constant.c1.value = YYYY
##
Equipment.constants = DEFAULT_CONTROL_STATE
Equipment.constant.DEFAULT_CONTROL_STATE.name = DefaultControlState
Equipment.constant.DEFAULT_CONTROL_STATE.value = 1

##
# Equipment Reports Definition
#
#Equipment.reports = r1 r2 r3 .....
#Equipment.report.r1.name = --- the name of the report
#Equipment.report.r1.rptid = --- the report id
#Equipment.report.r1.event = ev1 ---- the name of the collection event from the driver
#Equipment.report.r1.variables = v1 v2 v3 ---- a list of data variables that are contained in the
driver
#
#
##

##
# The Scenarios that this EM supports (space separated list)
# These can be in built or custom defined Scenarios that extend the Scenario class
##
Scenarios = LOAD UNLOAD

##
# The Operations that are configured for each scenario
```

```
# Each scenario must have at least one operation , either in built or custom defined
#
#Scenario.XXX.operations = op1 op2 op3 op4 // the operations that this scenario executes
#Scenario.op1.dependencies = var1 var2 // the variables that this scenario depends upon, must be
configured into Status Manager
#Scenario.var1.goodToGo = X Y Z // for each variable we indicate a good to go value for the
operation
#
##
Scenario.LOAD.operations = SELECT_RECIPE START
Scenario.UNLOAD.operations = UNLOCK

##
# Operations have the following properties available
#
# Operation.XXX.transactionName = : must be set to the name of a transaction defined in the
custom XML library
# Operation.XXX.eventName = : must be set to the name of an event in the custom XML library
appended with a timeout, transaction return
#           blocked until this event has been received or the timeout has been reached
# Operation.XXX.className = : A user developed class that extends the Operation class
# Operation.XXX.parameterList = : The list of parameters that the execute() method of the afore
#           mentioned class requires
#
# by default all parameters are taken as empty.
##

##
# Operation Mappings
##
Operation.SELECT_RECIPE.transactionName = SELECT_RECIPE

# no event after the start
Operation.START.transactionName = START

# no event after the unlock
Operation.UNLOCK.transactionName = UNLOCK

##
# The following section configures application events
#
# An ApplicationEvent is a message that is sent to the host application, based upon
```

```
# the outcome of an Operation or Reception of a particular Equipment Collection Event
#
# ApplicationEvents are used for things such as:
#
#           Informing the MES when a Equipment changes its state
#           Data Collection etc.
#
# The events configures are based upon the following properties
#
#ApplicationEvents = Event1 Event2
#ApplicationEvent.Event1.eventName = Process State Changed
#ApplicationEvent.Event1.operationName = the name of the operation that has completed
#
# only one of the above properties may be set for each ApplicationEvent, if no property is set then
no event is fired
ApplicationEvents = MBC_COMPLETE
ApplicationEvent.MBC_COMPLETE.eventName = Process State Changed

##
# The following section configures the equipment status monitor
# Each variable in the list is updated when the particular event is received
# or a status query with the variable in is made
#
#Status.variables = var1 var2 var3
#Status.var1.event = eventName
#Status.var2.event = eventName
#Status.var3.event = eventName
#
# where variableName is the name of the variable in the custom.xml library
# where eventName is the name of the event in the custom.xml file
##
Status.variables = ControlState ProcessState
Status.ControlState.event = Control State Changed
Status.ProcessState.event = Process State Changed

##
# The following section configures the AlarmManager
#
# if enableAll is set to true enableAlarms is disregarded, enableAll.transactionName provides
# the name of the transaction used to enable all the alarms
#
```

```
# The AlarmManager will not action any alarms from the ignoreAlarms list
#
#Alarms.enableAll = true/false
#Alarms.ignoreAlarms = alarm1 alarm2 .....
#Alarms.enableAll.transactionName = XXXX
#
# if enableAll is false, the list of alarms in enableAlarms is enabled, the transaction specified in
# enableAlarms.transactionName is used to achieve this.
#
#Alarms.enableAlarms = alarm1 alarm2 ...
#Alarms.enableAlarms.transactionName = XXXX
#
#
# The AlarmManager will ignore any alarms with any severity present in the ignoreSeverity list
#
#Alarms.ignoreServerity = severity1 severity2
#
# The AlarmManager will process primary messages from the following transactions as alarms
#
#Alarms.transactions = trans1 trans2 .....
#
##
```

Appendix – Example ‘XML Library’ File

```
<?xml version="1.0" encoding="UTF-8"?>
<library name="TESTTOOL" version="1.0">
  <SecsElements>
    <SecsElement fixed="true" format="Binary" name="ACKC5">
    </SecsElement>
    <SecsElement fixed="true" format="Binary" name="ACKC6">
    </SecsElement>
    <SecsElement fixed="false" format="Binary" name="ALCD">
    </SecsElement>
    <SecsElement fixed="false" format="U4" name="ALED">
    </SecsElement>
    <SecsElement fixed="true" format="Ascii" name="TIME">
    </SecsElement>
    <SecsElement fixed="false" format="Ascii" name="UNITS">
    </SecsElement>
    <SecsElement fixed="false" format="U4" name="V">
    </SecsElement>
    <SecsElement fixed="false" format="U4" name="VID">
    </SecsElement>
  </SecsElements>
  <CollectionEvents>
    <CollectionEvent ceid="1012" description="Recipe Complete" name="Recipe
Complete">
    </CollectionEvent>
    <CollectionEvent ceid="2012" description="Recipe Started" name="Recipe
Started">
    </CollectionEvent>
    <CollectionEvent ceid="444" description="Control state" name="Control
State Changed">
    </CollectionEvent>
  </CollectionEvents>
  <Reports>
    <Report name="0To be user defined" rptid="0">
      <CollectionEvent ceid="7777" description="process state" name="Process
State Changed">
        </CollectionEvent>
        <DataVariables>
        </DataVariables>
      </Report>
    </Reports>
  <StatusVariables>
    <StatusVariable svid="87" svname="Cassette Present" units="1">
    </StatusVariable>
    <StatusVariable svid="1234" svname="ControlState" units="17">
    </StatusVariable>
  </StatusVariables>
  <DataVariables>
    <DataVariable dvid="100" dvname="LOT_ID" units="">

```

```
</DataVariable>
  <DataVariable dvid="200" dvname="RECIPE_ID" units="">
</DataVariable>
</DataVariables>
<EquipmentConstants>
</EquipmentConstants>
<Alarms>
  <Alarm alcd="2" alid="16" altx="CH_A_High_Temp">
</Alarm>
  <Alarm alcd="1" alid="1554" altx="Chamber A Open">
</Alarm>
  <Alarm alcd="1" alid="1555" altx="Chamber_B_Open">
</Alarm>
</Alarms>
<Transactions>
  <Transaction name="New Equipment Constant Send">
    <Primary direction="toEquipment" function="15" generic="true"
mnemonic="ECS" multiblock="true" name="New Equipment Constant Send"
replyRequired="false" stream="2">
      <L2>
        <Item format="U4" name="ECID" value="0">
</Item>
        <Item format="U4" name="ECV" value="0">
</Item>
      </L2>
    </Primary>
    <Secondary direction="toHost" function="16" generic="true"
mnemonic="ECA" multiblock="false" name="New Equipment Constant Acknowledge"
replyRequired="false" stream="2">
      <Item format="Binary" name="EAC" value="0">
</Item>
    </Secondary>
  </Transaction>
  <Transaction name="UNLOAD">
    <Primary direction="toEquipment" function="41" generic="true"
mnemonic="HCS" multiblock="false" name="Host Command Send" replyRequired="true"
stream="2">
      <L2>
        <Item format="Ascii" name="RCMD" value="UNLOAD">
</Item>
        <L2>
          <L2>
            <Item format="Ascii" name="CPNAME" value="LOTID">
</Item>
            <Item format="Ascii" name="CPVAL" value="$MID">
</Item>
          </L2>
          <L2>
            <Item format="Ascii" name="CPNAME" value="PORT">
</Item>
            <Item format="Ascii" name="CPVAL" value="$LOCATION">
</Item>
          </L2>
        </L2>
      </L2>
    </Primary>
  </Transaction>
</Transactions>
</DataVariables>
</EquipmentConstants>
</Alarms>
</Transactions>
```

```
        </L2>
    </Primary>
    <Secondary direction="toHost" function="42" generic="true"
mnemonic="HCA" multiblock="false" name="Host Command Acknowledge"
replyRequired="false" stream="2">
    <L2>
        <Item format="Binary" name="HACK" value="0">
    </Item>
    <L1>
        <L2>
            <Item format="Ascii" name="CPNAME" value="PPID">
    </Item>
            <Item format="Binary" name="CPACK" value="0">
    </Item>
        </L2>
    </L1>
    </L2>
    </Secondary>
</Transaction>
</Transactions>
</library>
```