Simulation Suite V2.6

Getting Started

gemalto
security to be free

GEMALTO, B.P. 100, 13881 GEMENOS CEDEX, FRANCE.

Tel: +33 (0)4.42.36.50.00 Fax: +33 (0)4.42.36.50.90

Printed in France.

Document Reference: D1177488A

October 28, 2010

Contents

List of Tables

List of Figures

The Simulation Suite V2.6 packages everything a developer needs to simulate Wireless Solution. A real Swiss Army Knife if you need to test or demonstrate a full end-to-end solution, from card side to server side.

This document is designed to help you learn about Gemalto Simulation Suite V2.6 by using it.

Who Should Read this Book

You only need a minimum knowledge about smart cards, but you are strongly recommended to read the standards and documents listed in “References” on page 99.

Conventions Used in this Document

bold	Command and window names are shown in bold. For example: ... the **JCardManager** window...
>	Menu selection sequences are shown using the > symbol to link the selections. For example: ... select **Start > All Programs > Gemalto > Wireless Solutions**...
italics	Book titles are shown in *italics*.
notation	■ By default, a numeric value is expressed in decimal notation. ■ Whenever a value is expressed in binary, it is followed by the letter “b”. For example the decimal value 13 expressed in binary becomes **1101b**. ■ A hexadecimal number is followed by the letter “h”, or preceded by “0x”. For example, the decimal value 13 expressed in hexadecimal becomes **0Dh** or **0x0D**.

installdir	This variable is used throughout this book to indicate the directory in which Simulation Suite V2.6 is installed. For example, when the following directory is shown in the book: *installdir*`\bin` you may need to substitute the following directory: `c:\Program Files\Gemalto\SimulationSuite\bin`
carddir	This variable is used throughout this book to indicate the directory which stores the files common to a particular type of card. For example, when the following directory is shown in the book: `...\examples\`*carddir* you may need to substitute the following directory: `...\examples\USIM Card R6`

What’s New in this Guide

In this guide, the Simulation Suite has extended its feature to cover the following:

- Application Manager’s **Object Journalism Feature** provides the option to install application in high endurance area, in order to extend the overall lifetime of the flash memory.
- Direct access to **Crypto Tool** and **OTA Interpreter** on the Simulation Suite’s tool bar, which these only able to access through Developer Suite previously.
- **OTA Configuration Interpreter and Editor** which allows the user to configure the ota.conf file in card to set the SCWS remote administrative parameters.
- **OMA SCWS Server Simulator** which provides the user a simulated administration protocol environment to remotely administrate the SCWS.

Contact Our Hotline

If you do not find the information you need in this manual, or if you find errors, contact the Gemalto hotline at http://support.gemalto.com/.

Please note the document reference number, your job function, and the name of your company. (You will find the document reference number at the bottom of the legal notice on the inside front cover.)

Introduction

Simulation Suite V2.6 Overview

A Full End-To-End Simulation Tool

The Simulation Suite V2.6 combines a standalone End-to-End simulation environment and a project management tool to facilitate your Java Card project test.

- Standalone End-to-End Simulation Environment

  The Simulation Suite provide all components needed to perform an end-to-end simulation of Wireless Solutions.

  - Card Simulators
  - Mobiles Simulators
  - Network Simulators (with Card Profile Management)
  - Server Simulators

  Also it provides the components to manage the Files and Applications in the Card:

  - Application Manager/JCardManager - manage Applets and Servlets on both Real Cards and Card Simulators
  - Static Content Explorer - manage Static Content in the SCA Memory of either a real SCWS Card or a SCWS Card Simulator
  - File System Editor - basic management of DF/EF on a Real Card or a Card Simulator

- Project Management

  The Main Use Case of the Simulation Suite is to run a Project that has been created using the Developer Suite.

For this, the Simulation Suite provides the possibility to import Projects - either with or without source code (referred as a Binary Project) and run it without changing any configuration.

Developer Suite JavaCard Project Management

The Simulation Suite V2.6 contains all tools you use to demonstrate or test Developer Suite Projects. It imports wireless solutions you developed with and exported from the Developer Suite V3.2 or later, and works independently of the development environment.

Note: Only Developer Suite V3.2 or higher supports the project export feature.

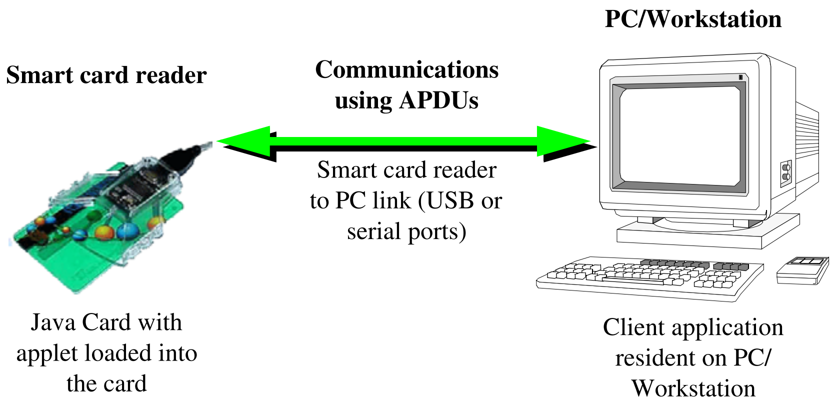
Before you start using the Simulation Suite V2.6, some basic concepts of Java Card Environment and background information about Java Card Project development process using Developer Suite may help you better understand the Simulation Suite V2.6 functionalities and its role in the whole development cycle.

If you are familiar with Java Card development and the Developer Suite developing tool, you can skip this section.

The Java Card Environment

"Figure 1" shows the main components of a Java Card application.

Figure 1 - The JavaCard Environment: a PC/Workstation-Resident Client Application


PC/Workstation
Smart card reader
Communications using APDUs
Smart card reader to PC link (USB or serial ports)
Java Card with applet loaded into the card
Client application resident on PC/ Workstation

Java Cards contain a microprocessor, but have no means of directly receiving input or displaying output. They must therefore be connected to a card reader, also known as a card acceptance device (CAD), or terminal, which is in turn connected to or contained in:

- A workstation such as a personal computer (PC) via a USB or serial cable.
- A mobile phone
- Any other smart card reading device.

The card in a card reader and the client application communicate using application protocol data unit (APDU) commands. APDUs contain command instructions, data and responses. Refer to the *ISO 7816-3* standard for detailed information on APDUs.

The GlobalPlatform Specifications

The GlobalPlatform (GP) is a generic framework for the management of multi-application smart cards, devices and systems. It provides mechanisms for securely managing the applications on smart cards in order to extend the Java Card environment.

Note: The GlobalPlatform specification was formerly named Open Platform. OP 2.0.1 and GP 2.0.1 are exactly the same apart from the name.

USIM R6 and R-UIM cards are GP 2.1.1 compliant. The other members of the GemXplore card family are fully compatible with the GP 2.0.1 standard.

The GP specification includes several components:

- A set of commands to manage the life cycle of the card and its applications, load, install, and delete the applications on the card, and manage the card's security by, for example, updating keys and setting up a secure channel between the card and the terminal.
- An API, consisting of a single Java package, `org.globalplatform` (or `visa.openplatform` in the case of GP 2.0.1) that can be used by application developers to access the GP features, in particular the application's life cycle and the secure messaging mechanism.
- A documented specification, which describes in detail the commands available and the principles of interoperability between the Java Card and GP card environments. The current version is GP 2.1.1.

To obtain the GP specification, go to www.globalplatform.org.

The OpenCard Framework

The OpenCard Framework (OCF) is an open architecture and a set of APIs providing a common interface for card readers and smart card-based applications. The OCF enables you to develop client applications that will work on a variety of different suppliers' equipment and platforms. The Developer Suite V3.6 tools use the OCF for all communications between client applications, card readers, real cards and simulated cards. An OCF to PCSC bridge is used to be able to manage PCSC card readers in OCF.

To obtain further details about the OpenCard Framework, go to www.opencard.org.

Java Card Concepts and Terminology

Applets

An applet is a Java program designed to work within the Java Card environment. Apart from being programmed in Java, a Java Card applet is very different from a web browser-based Java applet.

Note: An applet is often referred to in specifications as a "Java Card application". The GlobalPlatform documentation, for example, is not specific to Java Cards and uses the term "application" throughout. You can also load packages with no applets onto cards, and applets can have instances, so "application" is a generic term.

Applets can be pre-installed on a card during the manufacturing process. They can also, however, be downloaded to a card at subsequent stages of the card's life cycle. On GSM cards, for example, applets can be downloaded:

- From the wireless network using "over the air" techniques.
- At the point-of-sale.

Many different applets can be installed on the same card, and multiple instances of the same applet can be created ("instantiated") and run on the same card. You can also load packages containing library functions that are used by other applets on the card.

In the telecommunications environment, Java Card applets are server applications running in the card. These applets react to the mobile equipment (ME) user interface or to an OTA application.

A telecom Java card can manage three types of API:

- Toolkit
- Java Card
- GlobalPlatform

All GemXplore cards can support all three types of API.

Applets written for telecom applications can use the SAT (SIM Application Toolkit), CAT (Card Application Toolkit), USAT (USIM Application Toolkit) or CCAT (CDMA Card Application Toolkit) APIs. These applets must be created using the appropriate Toolkit Applet wizard.

Applets written for Java Card applications can be written using either exclusively Java Card API methods (Java Card applets), or a combination of Java Card and GlobalPlatform methods (Java Card-GP applets). Both types of applets must be created using the Java Card Applet wizard. The choice of Java Card or Java Card-GP is made during step 4 of the Java Card Applet wizard by checking (or not) **Global Platform specifications,** as shown on.

GSM and 3G Extension to the Java Card Specification

In order to support the SIM Toolkit mechanisms used by applications on GSM cards, the Java Card API has been extended with a GSM-specific API.

This API is defined in ETSI 102–241 (a common Toolkit API for Telecom technologies) and either the 3GPP TS 43.019 (for 2G cards) or 3GPP 31–130 specification (for 3G cards). Very basically, it enables applets to access the file system on a SIM card, and to perform proactive exchanges with a mobile as defined in the SIM Toolkit specification.

CDMA Extensions

Developer Suite V3.6 provides support for CDMA target cards, allowing you to generate skeleton code for CCAT (CDMA Card Application Toolkit) applications.

You can use Developer Suite V3.6’s Java Card project and applet creation wizards to generate CCAT-compatible skeleton code for your project, then test and simulate application deployment using Developer Suite V3.6 tools such as the Mobile Simulator CDMA and the script editor.

The R-UIM card profile complies with the following specifications:

- 3GPP2 CS0035-A (Version 1.0)
- ETSI TS 102.223 (CAT)
- ETSI TS 102.241 (UICC API)
- Java Card 2.2.1

Client Applications

A client application is a software program running on a workstation. The client application communicates with an applet that has been uploaded and installed on a card.

The client application controls all input and output operations on behalf of the applet, and sends requests for processing to the applet. The applet acts as a server, responding to the client application’s requests.

The following client applications are provided with Developer Suite V3.6:

- The Mobile Simulator GSM, Mobile Simulator 3G, and Mobile Simulator CDMA.
- The JCardManager.
- Simulation Chain 2G and Simulation Chain 3G.
- SATSA Midlet

You can also develop your own custom client applications with Eclipse. Developer Suite V3.6 provides you with wizards to help you do this. To obtain further details about developing your own client applications, see the documentation available at www.opencard.org.

Targets

Applets can be loaded onto two types of target:

- A real Java Card, which can be either ETSI 102-224/102-225/102-226 compatible, or GlobalPlatform compatible.
- A card simulator supplied with Developer Suite V3.6, which simulates a card/card reader combination. The card simulator makes it possible to develop applets without having access to real Java Cards or a card reader and also allows you to debug your applets.

Packages

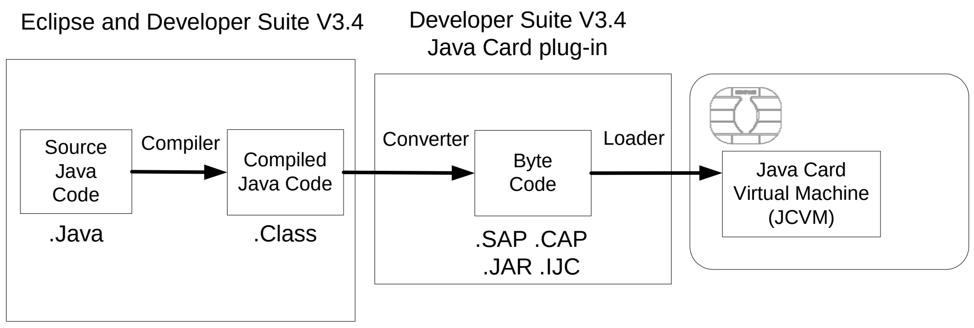
A package is a Java programming term to describe the collection of related classes and interfaces to which an applet belongs. Since an entire package can be uploaded to a card, it is easy to upload several applets at the same time. If you choose to develop your own client application, the client application’s classes must belong to a different package than the applet’s classes. A package can also be created that only contains library classes to be loaded, that is, no applets.

In global system for mobile communications (GSM) environments, a package is often referred to as an “executable load file”.

File Formats

Before an applet can be loaded onto a card, it must be converted to byte code that can be interpreted by the Java Card Virtual Machine (JCVM) on the card. The process is illustrated in “Figure 2 - The Compilation, Loading and Conversion Process”:

Figure 2 - The Compilation, Loading and Conversion Process



The process involves converting the Java source files first to compiled Class files, then to byte code. The byte code is then loaded onto the target card, where it is interpreted by the Java Card Virtual Machine.

CAP Files

The “.cap” file format is the **loadable file format** defined by the Java Card specification. It defines the binary representation of an applet (or library for packages with no applets) that can be loaded and executed on a Java Card. A “.cap” file consists of a set of components, each of which describes a different aspect of the contents. The set of components in a CAP file includes a number of mandatory components defined in the Java Card specification, but may also include a number of optional components that are not necessarily supported by all card manufacturers.

For a complete description of the Java Card CAP file format, see Sun’s *Java Card 2.2.1 Virtual Machine Specification*.

JAR Files

The “.jar” file structure is also defined in the Java Card specification. A “.jar” file **is NOT a loadable file format**, it is simply an archive (similar to a “.zip” archive) containing a set of “.cap” files, each “.cap” file corresponding to one loadable component. In order to be loaded on a card, a “.jar” file must systematically be converted into a loadable “.cap” file (this very basically consists in concatenating the individual “.cap” components from the “.jar” archive).

SAP Files

Simulator Applet Package (SAP) files are a Gemalto-proprietary version of the CAP files. Although they cannot be downloaded onto a real card, when loaded into Developer Suite V3.6’s card simulator they accurately simulate an applet in almost all other ways.

IJC Files

The “.ijc” format is not strictly speaking a standard file format. It was created by the SIM Alliance™ because the converter provided by Sun only generates JAR files, leaving JAR to CAP conversion open, which caused divergences in CAP file formats from one manufacturer to another.

The idea was thus to define a loadable file format containing only the components described as mandatory by the Java Card specification (for example, the optional “Descriptor” component has been removed in the IJC format).

In other words, the IJC format is the smallest common denominator of the different CAP file components supported by the cards of the SIM Alliance members.

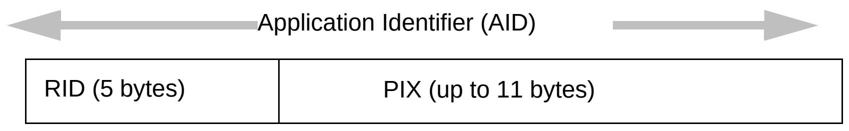
Projects

The term “project” in the Developer Suite environment refers to a package which may, or may not, include one or more applets, together with all necessary system classes, with or without libraries. When no applets are included in the project it is a library. The “project file” groups together all the files and environment variables required for building, debugging and loading applets and libraries.

AIDs

For security reasons, client applications must be able to uniquely identify any installed applet. Every package, applet, and instance of an applet loaded on a card must therefore be assigned a unique identifier, known as an application identifier (AID). An AID is a string of between 5 and 16 hexadecimal bytes.

Figure 3 - The AID Format



The first five bytes of an AID (the RID) indicate the application provider, that is, the company supplying the package or applet. To obtain an RID, your company must register with the international standards organization (ISO). This procedure is described in ISO7816-5 “Numbering system and registration procedure for application identifiers”.

The remaining bytes (up to 11) of an AID contain the proprietary identifier extension (PIX). The PIX is chosen by the application provider to uniquely identify a package or applet. Your company is then responsible for assigning PIXs to individual applications.

As mentioned previously, three types of AID are used in a project:

- The package AID. An AID assigned to a package.
- The applet AID. An AID assigned to an applet.
- The applet instance AID. An AID assigned to an instance of an applet.

An applet AID cannot have the same value as the AID of any other package or any other applet stored on the same card.

The Developer Suite V3.6 Tools

The Developer Suite V3.6 provides all components needed during each development status.

- **Quick Start Tool:** The Java Card Project Wizards gather the information necessary to build and configure a skeleton Java Card project, such as Java package names and default applet AID values.
- **Pre-loading tools**: The Developer Suite Java Card plug-in for Eclipse prepares the applet for loading onto the card by converting Class files to byte code.
- **Loading tools and post-loading tools**:
  - The JCardManager loads packages into a card or the card simulator, and exchanges APDU commands with a loaded applet.
  - The Application Manager loads and installs applets into a card.
  - The Script Editor creates and edits XML script files to provide batch execution of commands.

- **Simulation tools**: A card simulator simulates all aspects of particular card type's behavior, and is extremely useful for testing and debugging the functionality of an applet. The Mobile Simulator GSM, Mobile Simulator 3G and Mobile Simulator CDMA card simulators operate in exactly the same way as the equivalent real mobile telephones. The Simulation Chain 2G and Simulation Chain 3G tools simulate a secure connection between a server and a mobile station.
- **Miscellaneous utilities**: Various utility tools are also included:
  - The Cap File utility lets you visualize the contents of a loadable CAP file.
  - The Crypto Tool provides a method of enciphering and deciphering data using the DES, 3DES, AES and RSA cryptographic algorithms and provides a method of hashing using the SHA–1 and MD5 hash algorithms.
  - The File System Editor allows you to view the file structure in a card or card simulator and the properties and access conditions for any given file.
  - The Key File Editor allows you to create custom key files.
  - The OTA interpreter allows you to display an "interpreted" view of a sequence of OTA data bytes.
  - The WTK–OCF Bridge assures communication between the J2ME Mobile Simulator and the SmartCardFramework. It converts the TLP224 WTK2.3 mobile simulator protocol into the TPDU format used by OCF.

The following table lists the Developer Suite V3.6 tools and the corresponding development stages and target types required. All rows with **light-grey** background are also member components of the Simulation Suite V2.6.

Table 1 - Development Stages and Steps (continued)

Development Tools	Development Stages
Quick-start tools:	
Wizards	Code development and compilation.
Pre-loading tools:	
Developer Suite V3.6 Java Card Plug-in for Eclipse	Converting Class files to loadable files (CAP or SAP).
Loading tools and Post-loading tools:	
Application Manager	Loads and installs applets.
JCardManager	CAP file loading, APDU command exchange, tracing, scanning.
Script Editor	Creation and edition of configuration files and scripts.
Simulation and test tools:	
Simulation Chain 2G	Simulates a client application, GSM network components and a mobile phone.
Simulation Chain 3G	Simulates a client application, 3G network components and a mobile phone.
Mobile Simulator GSM	Simulates a client GSM handset.
Mobile Simulator 3G	Simulates a client 3G handset
Mobile Simulator CDMA	Simulators a client CDMA handset
JCardManager	APDU command exchange, tracing, scanning.

Table 1 - Development Stages and Steps (continued)

Development Tools	Development Stages
Card simulator	Testing and debugging applets
Miscellaneous utilities:	
Cap File utility	Viewing the contents of a CAP file
Key File Editor	Creating custom key files
CryptoTool	Enciphering and deciphering data
OTA Interpreter	Displays an “interpreted” view of a sequence of 23.048-formatted data bytes
File System Editor	Viewing the file structure and individual properties and access conditions for a file.
WTK–OCF bridge	A bridge between Sun’s Wireless Toolkit (WTK) phone simulator and Developer Suite V3.6's reader/card connection. Enables communications with a real reader/ card connection (since the WTK’s TLP224 protocol is integrated directly in the card simulators).

From “Table 1” we can see that Simulation Suite V2.6’s features overlap a major portion of the Developer Suite’s tools in loading, post-loading as well as test stages.

Developer Suite V3.6 Development Scenarios

The Developer Suite V3.6 tools allow you a wide range of flexibility when developing applets. There is usually more than one way of accomplishing a task.

Development Phase

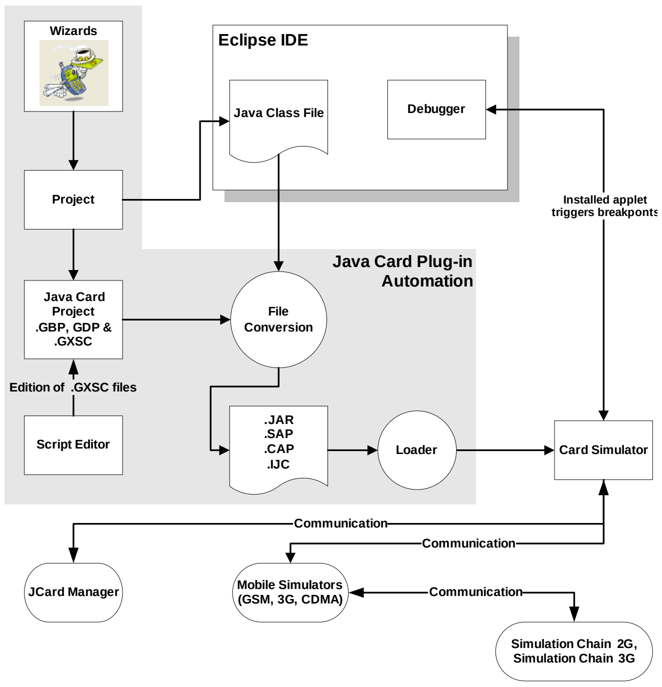
For the rapid development of a prototype applet with Developer Suite V3.6, do the following:

- **Run the Wizards**. Use the supplied wizards to generate skeleton code and a project definition.
- **Debug the Project**. Set breakpoints and then use Developer Suite V3.6 tools to launch the simulator, load the files into the card simulator and launch the client application.

Developer Suite V3.6 automatically generates a GXSC script file containing all necessary load and install parameters. If necessary, you can edit this file using the **Script Editor**.

The following diagram illustrates the development phase.

Figure 4 - The Development Phase



Tools Used During the Test Phase

Once you have built and debugged your Java Card application, it can be further tested in a real card.

The test scenario would be:

- Load the applet into the card with JCardManager (possibly using GXSC scripts) or the Application Manager (refer to “Using the Application Manager” on page 47). Alternatively, you can use the deployment script generated with the project.
- Test the applet using the client applications provided with Developer Suite V3.6:
  - The Mobile Simulator GSM, Mobile Simulator 3G, or Mobile Simulator CDMA
  - Simulation Chain 2G or Simulation Chain 3G
  - JCardManager (for Java Card applets)
  - OCF Client application (for Java Card applets)
  - JCRMI Client application (for JCRMI applets)
  - SATSA Midlet

Installing Simulation Suite V2.6

This chapter describes how to install Simulation Suite V2.6, including installation prerequisites, installation steps and post installation instructions.

System Requirements

Hardware Requirements

The following hardware is required to install and use Simulation Suite V2.6:

- A Windows-compatible PC with:
  - Intel® Pentium® 4 1.66 GHz (or faster).
  - 1 GB RAM recommended minimum (2 GB is recommended).
  - 1.7GB of available disk space, plus another 1GB during installation.
  - Support for Super VGA resolutions (1280 x 1024 is recommended).
- A PC/SC compliant smart card reader with associated connection cables and drivers, such as the GemPC Twin smart card reader (*P/N HWP108765 C*); this is not required for the Evaluation Version.
- Optionally, a SIM Toolkit, USIM Toolkit, or CCAT (CDMA Card Application Toolkit) Java Card (for example, GemXplore Xpresso v3 or GemXplore Generations). In order to be able to download applets onto a card in Simulation Suite V2.6, it must be either Global Platform or 3GPP 23.048 compliant.

Software Requirements

1. Simulation Suite V2.6 runs under the following operating systems:
   - Microsoft® Windows XP (SP3)
   - Microsoft Vista (SP1)
2. The Java™ 2 Runtime Environment (JRE) 1.5 or higher. Java JRE 1.6 is installed as part of the Simulation Suite V2.6 installation procedure.
3. Java Cryptography Extension, necessary in order to use Simulation Chain 2G and Simulation Chain 3G to their full capabilities (for example, allows the use of 24-byte 3DES keys). To fully benefit from all the features of Simulation Chain 2G and Simulation Chain 3G, you should use the “unlimited strength” version. These can be downloaded from http://java.sun.com/products/JCE.

4. Adobe® Acrobat® Reader Version 4.0 or later. This is required to read the online documentation provided on the installation CD-ROM.
5. Microsoft Internet Explorer Version 5.01 and Microsoft XML Parser Version 4.0, or Microsoft Internet Explorer Version 6.0 or later.
6. If you are using the card simulator, you must make sure the TCP/IP networking protocol is installed on your PC.

Installation Overview

You can either install the Simulation Suite V2.6 environment from a `.zip` file downloaded from the product website. The installation program installs any additional components required. If you have a card reader installed, it is configured for use. The card simulator is automatically installed on your workstation during installation.

Before Installing Simulation Suite V2.6

To prepare for Simulation Suite V2.6 Installation, perform the following steps:

1. Configure the display settings on your computer. The desktop resolution must be at least 1,024 by 768 pixels, with 65,536 colors (16–bit).
2. Log in with an account that has Administrator rights on your computer, otherwise you will not be able to install Simulation Suite V2.6. Contact your System Administrator if do not have these rights.
3. Close all other applications running on your computer.

Installing Simulation Suite V2.6

To install Simulation Suite V2.6:

1. Insert the Simulation Suite V2.6 CD-ROM into the CD-ROM drive on your computer; or unzip the installation archive (the `.zip` file) to a temp directory and double-click the executable program (like `Installer.exe`).

   The installation program starts automatically.

   For CD-ROM installation, if for any reason it does not start automatically, run the ***drive*:\Simulation_Suite_*x.y.z*.exe** program on the installation CD-ROM, where *drive* is the drive letter of the CD-ROM drive.
2. Choose the **Install Simulation Suite V2.6 Installation** option. This program guides you through the installation process.

Note: The default software installation directory is:

```
c:\Program Files\Gemalto\Simulation Suite
```

You can specify a different drive or directory. To avoid confusion, the installation directory is referred to as “*installdir*” throughout this document.

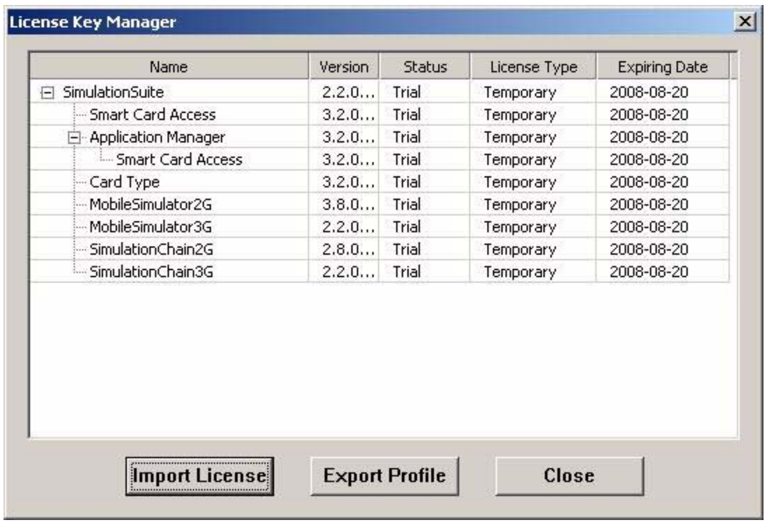
After Installing Simulation Suite V2.6

Apply and Import A License for Simulation Suite V2.6

When installation of the software is complete:

1 **License Key Manager** window will prompt, showing you the license information. This window will be removed after you register the Simulation Suite product.

Figure 5 - License Key Manager



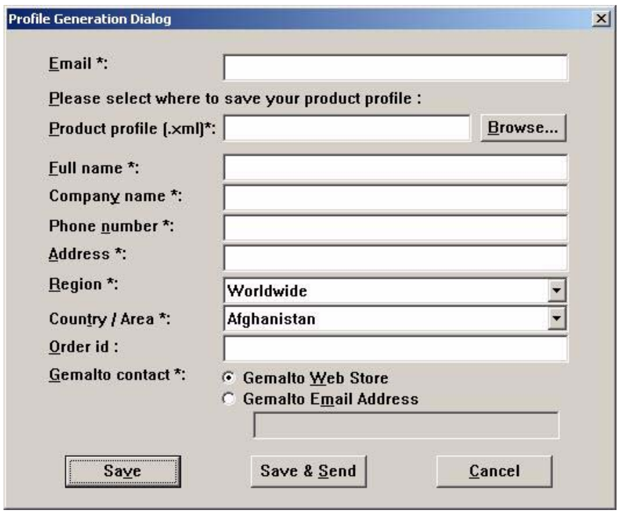
License Key Manager

Name	Version	Status	License Type	Expiring Date
SimulationSuite	2.2.0...	Trial	Temporary	2008-08-20
Smart Card Access	3.2.0...	Trial	Temporary	2008-08-20
Application Manager	3.2.0...	Trial	Temporary	2008-08-20
Smart Card Access	3.2.0...	Trial	Temporary	2008-08-20
Card Type	3.2.0...	Trial	Temporary	2008-08-20
MobileSimulator2G	3.8.0...	Trial	Temporary	2008-08-20
MobileSimulator3G	2.2.0...	Trial	Temporary	2008-08-20
SimulationChain2G	2.8.0...	Trial	Temporary	2008-08-20
SimulationChain3G	2.2.0...	Trial	Temporary	2008-08-20

2 If a license is already achieved, click **Import License** to browse and locate it on your local disk. Follow the wizard to complete.

3 If you have no license on hand, you need to apply one first. Click **Export Profile**, the following window appear:

Figure 6 - Profile Generation



Profile Generation Dialog

Email *:	
Please select where to save your product profile :	
Product profile (.xml)*:	
Full name *:	
Company name *:	
Phone number *:	
Address *:	
Region *:	Worldwide
Country / Area *:	Afghanistan
Order id :	
Gemalto contact *:	[✓ Gemalto Web Store] Gemalto Email Address

4 Fill in the blank form in window as the above figure presents. All fields with a * are required. For the Gemalto contact option, choose either the “Gemalto Web Store“ or “Gemalto Email Address“ as your email receiptor.

5 Click **Save** if you want to further revise your information before sending; click **Save & Send** to take modification in effect and feedback them to Gemalto.

6 A message will the be sent to your registered Email Box. A license (XML file) will be attached.

7 Go back to the “License Key Manager” window. Click **Import License** and follow the wizard to complete.

Simulation Suite V2.6 Components

The Simulation Suite V2.6 provides all components needed to perform an end-to-end simulation of Wireless Solutions, as well as those to manage the Files and Applications in the card:

- Project Manager - import and export a Developer Suite Project with last running data.
- Simulators
  - Card Simulators
  - Mobile Simulators
  - Network Simulator (with card profile management)
  - Server Simulator
- Card File Management Tools
  - Application Manager 2G/3G - manage Applets and Servlets on both Real Cards and Card Simulators.
  - File System Editor - basic management of DF/EF on a Real Card or a Card Simulator.
  - JCard Manager - manage the use of a Java Card or mobile simulator target within a JDK environment.
- Static Content Explorer - manage Static Content in the SCA memory of either a real SCWS Card or a SCWS Card Simulator.
- Universal Tracer - enables you to trace APDU communication exchanges between a PC application and a PC/SC-compatible smartcard reader or card simulator to debug or check compliance with established standards.

Project Manager

The Project Manager helps import a Developer Suite project and run it without the Developer Suite development tool installed.

For this purpose, the Project Manager provides the possibility to import projects - either with or without source code (referred to as a Binary Project) and run it without changing any configuration. The project can either be an exported Developer Suite Archive (.ZIP file) or a directory which contains the project.

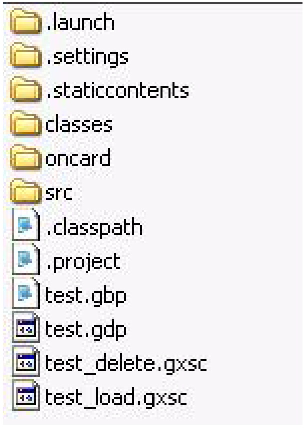
To run a Developer Suite project in Simulation Suite, you needn’t know how the project was exported from Developer Suite V3.6; the common practice is to specify the target Developer Suite Project’s location and import it, easy but enough. However, in case you need, please refer to *Developer Suite V3.6 Getting Started* (Developer Suite V3.6 is the most recent version that supports project import and export).

Developer Suite Project Contents

A Developer Suite Project is a Java Card Project you create with Developer Suite IDE. Developer Suite itself contains loading tools, simulation tools and other utilities used to debug or test a developed application. The Simulation Suite packages that part of functions and enables testing applications in a remote non-Developer Suite environment.

In “Figure 7” is a Developer Suite Project we create from an SCWS Project sample - “barcode” in Developer Suite V3.6. It represents the general structure of an exported Developer Suite Project:

Figure 7 - Developer Suite Project Contents


.launch
.settings
.staticcontents
classes
oncard
src
.classpath
.project
test.gbp
test.gdp
test_delete.gxsc
test_load.gxsc

- **.launch**: This folder contains the project launch file, which defines launch configuration.
- **.settings**: This folder contains project properties information, including Project Type in terms of the Card Profile it specified,
- **.stationcontents**: This folder only exists in an SCWS project. It contains oncard static contents which can be launched with the Static Content Explorer.
- **classes**: Under this folder are compiled files of your Java Card application. “class” is the extension of the source files.
- **oncard**: Converted byte code files are contained in this folder. These files are converted in development phase in order to be loaded and used in card. Possible byte code file formats are CAP, SAP, IJC, and JAR.

  Please refer to “File Formats” on page 5 for more information.
- **src**: Under this folder are source files of your Java Card application. “java” is the extension of the source files.
- **.classpath**: This file defines the project class path configuration, including source folder, export folder, and external referenced libraries’ directory.
- **.project**: This is the project definition.
- **test.gbp**: The project build file.
- **test.gdp**: The project debug file.
- **test_delete.gxsc**: Script used to delete applet from card.

- **test_load.gxsc:** Script used to load applet into card.

License Key Manager

The License Key Manager applies a license protection mechanism to the Simulation Suite V2.6.

It automatically manages the product license and under circumstance of license absent, user will only be able to use a trial version. After license expires, the Simulation Suite V2.6 and all its sub-components can not be launched.

It also gives users guide to registration. Each time a trial version of the Simulation Suite V2.6 is started, user will be prompted with a dialogue, asking for software registration, like “Apply and Import A License for Simulation Suite V2.6” on page 15 describes.

Universal Tracer

The Universal Tracer enables you to trace APDU communication exchanges between a PC application and a PC/SC-compatible smartcard reader or card simulator to debug or check compliance with established standards. When used with a special USB mobile-SIM connector (a MSC device), the Universal tracer can also track the APDU exchanges between a mobile equipment (ME) and a (U)SIM card. These communication exchanges are subsequently logged and interpreted.

Static Content Explorer

The static Content Explorer provides a visual viewer to manage static content in SCWS card. It is easy to use this tool to add new file (even batch file is allowed), delete folder/ files and explore contents. For more details about the Static Content Explorer, please refer to “Chapter 5 - The Static Content Explorer”.

JCardManager

The JCardManager is a powerful tool used to load CAP files, exchange APDUs, and run traces. More generally, it is used to send any Global Platform command to a card or simulator. For more details about the JCardManager, refer to “Appendix 6 - The JCardManager”.

Application Manager 2G/3G

The Application Manager is a tool that can be used to download packages onto real cards and install and delete applets or applet instances. It has three download modes:

- I/O mode: GlobalPlatform–compatible cards
- OTA mode: 3GPP TS 23–048 compatible cards
- BIP mode: 3GPP TS 23–048 compatible cards

The Application Manager is started from the **Developer Suite** menu in Eclipse IDE and is fully explained in “Chapter 7 - Using the Application Manager”.

Simulators

There are three types of simulators available in Simulation Suite V2.6:

- Card simulators (SIM, USIM, R-UIM and SCWS cards).
- Mobile simulators (Mobile Simulator GSM, Mobile Simulator 3G and Mobile Simulator CDMA).
- Simulation Chain 2G and Simulation Chain 3G.

The Card Simulators

Card simulators are software that simulate the behavior of a smart card. Simulation Suite V2.6 includes a number of card simulators. They perform the same basic functions, but are compliant with different versions of Java Card, Telecom and GlobalPlatform standards. The following table illustrates the differences:

Table 2 - Simulator Versions — Standards Compatibility

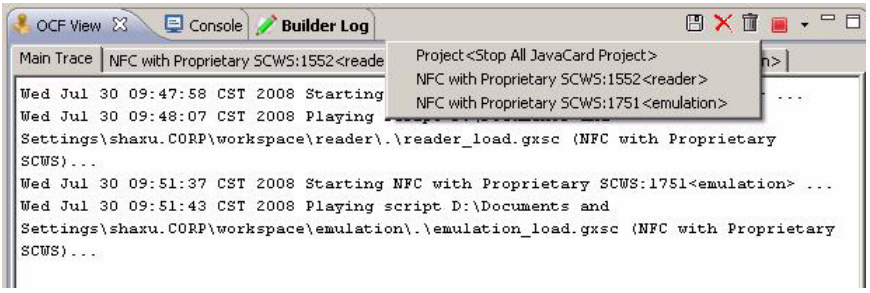
Simulator	**Java Card version**	**Telecom standard**	**GP standard**
USim Card Simulator R6	2.2.1	3GPP release 6	2.1.1
USim Card Simulator R5	2.2.1	3GPP release 5	2.0.1′
Sim Card Simulator R5	2.1.1	3GPP release 5	2.0.1′
Sim Card Simulator R99	2.1.1	3GPP (1999 release)	2.0.1′
R-UIM Card Simulator	2.2.1	3GPP2 C.S0035-A	2.1.1
SCWS Card Simulator	2.2.1	3GPP release 6	2.1.1

To start the SIM Card Simulator:

From the **Start** menu, choose **USim Card R6 Simulator, USim Card R5 Simulator**, **Sim Card R5 Simulator**, **Sim Card R99 Simulator**, **R-UIM Simulator** or **SCWS Card Simulator** as appropriate. If you cannot find the simulator directly from the **Start** menu, try **Start** > **All Programs** > **Gemalto** > **Wireless Solutions > Card Simulator.**

Alternatively, when a Java Card applet project is open in Eclipse, running the project automatically launches the simulator and loads the applet into the simulator. This is typically the method used if you want to debug an applet. This action displays the following in the Messages area of the Eclipse window:

Figure 8 - Sim Card Simulator Window


OCF View
Console
Builder Log
Main Trace
NFC with Proprietary SCWS:1552<reade
Project<Stop All JavaCard Project>
NFC with Proprietary SCWS:1552<reader>
NFC with Proprietary SCWS:1751<emulation>

```
Wed Jul 30 09:47:58 CST 2008 Starting
Wed Jul 30 09:48:07 CST 2008 Playing
Settings\shaxu.CORP\workspace\reader\.\reader_load.gxsc (NFC with Proprietary
SCWS)...
Wed Jul 30 09:51:37 CST 2008 Starting NFC with Proprietary SCWS:1751<emulation> ...
Wed Jul 30 09:51:43 CST 2008 Playing script D:\Documents and
Settings\shaxu.CORP\workspace\emulation\.\emulation_load.gxsc (NFC with Proprietary
SCWS)...
```

Card Simulator Secret Codes

Table 3 - Card Simulator Secret Codes

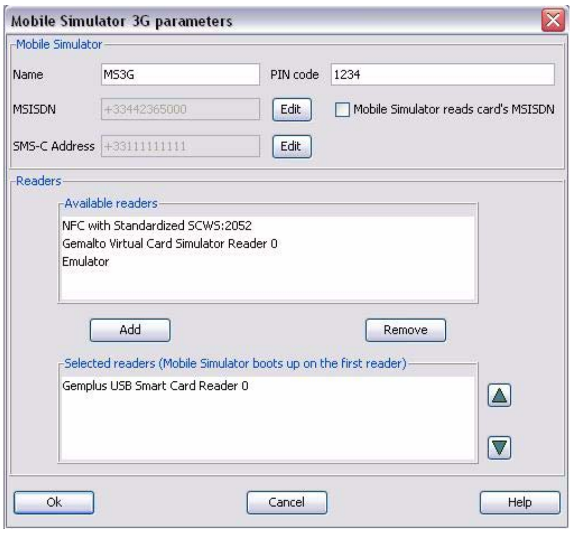
Simulator	Secret code	Code value	Unblock code value
SIM R99/R5 Simulators	CHV1	1234	11111111
	CHV2	5678	22222222
	ADM1	ADM_AUTH	-
	ADM3	ADM33333	-
USIM R5 Simulator	CHV1	1234	11111111
	CHV2	5678	22222222
	PIN1	1234	11111111
	ADM1	ADM11111	-
	ADM2	ADM22222	-
	ADM3	ADM33333	-
	ADM4	ADM44444	-
R-USIM Simulators	CHV1	1234	11111111
	CHV2	5678	22222222
	PIN1	1234	11111111
	PIN2	1345	11111111
	PIN3	1456	11111111
	PIN4	1567	11111111
	ADM1	ADM11111	-
	ADM2	ADM22222	-
	ADM3	ADM33333	-
	ADM4	ADM44444	-
NFC with SCWS and USIM R6 Simulator	CHV1	1234	11111111
	CHV2	5678	22222222
	PIN1	1234	11111111
	PIN2	1345	11111111
	PIN3	1456	11111111
	PIN4	1567	11111111
	ADM1	ADM11111	-
	ADM2	ADM22222	-
	ADM3	ADM33333	-
	ADM4	ADM44444	-

The Mobile Simulators

The mobile simulators are client applications used to debug Toolkit Applets. The simulators are all started in the same way. Each has a distinctive graphical user interface (GUI). The mobile simulator starts automatically when you run a project, provided you have selected it in the **Choose the client application to run** drop-down menu in **Run** or **Debug** window (see). Otherwise, you can start it manually from the **Developer Suite** menu by choosing the option **Mobile Simulator**.

The mobile simulators first display the Mobile Simulator Parameters window, as shown in the following figure:

Figure 9 - The Mobile Simulator Parameters Window



Mobile Simulator 3G parameters

Mobile Simulator

Name: MS3G
PIN code: 1234
MSISDN: +33442365000 Edit
☐ Mobile Simulator reads card's MSISDN
SMS-C Address: +33111111111 Edit

Readers

Available readers
NFC with Standardized SCWS:2052
Gemalto Virtual Card Simulator Reader 0
Emulator

Add Remove

Selected readers (Mobile Simulator boots up on the first reader)
Gemplus USB Smart Card Reader 0

Ok Cancel Help

Proceed as follows:

1 Select the reader or the card simulator from the **Available Readers** group box.

2 Click **Add** to add it to the **Selected readers** group box.

3 Click **OK** to continue.

Simulation Chain 2G and Simulation Chain 3G

This simulator is used to debug SIM Toolkit, USIM Toolkit, R-UIM Toolkit applets and Servlet that trigger OTA events. If you have it installed on your machine, you can start it in Eclipse from the **Developer Suite** menu by choosing the option **Simulation Chain 2G** or **Simulation Chain 3G**.

Smart Card Reader Manager

The Gemalto Smart Card Reader Manager is used to manage the smart card reader device in the PC which support protocol switching among the protocols, such as, ISO [T=0], ISO [T=1], ISO [T=2] and IC-USB [T=15]. Hence, it allows you to:

- Know the current protocol being used.
- Switch to another protocol that is supported.
- Know the state of the reader after protocol changes.

All these can be done by simply launching the Smart Card Reader Manager. To know more about how to use this application, please refer to "Chapter 10 - The Smart Card Reader Manager" for more details.

Card File Editing Tool

File System Editor

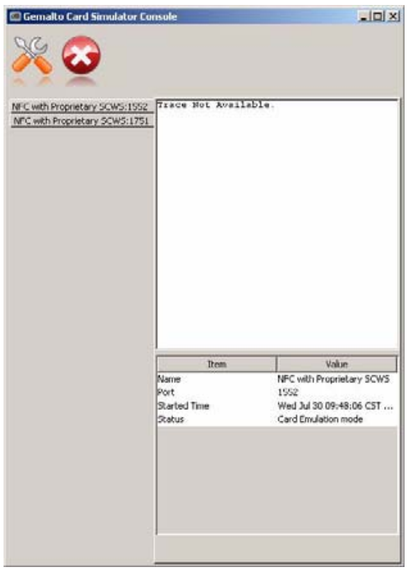
This tool enables you to view the file structure in a card or a card simulator and the properties and access conditions for any file. In addition it can be used to perform certain commands on files.

For information about this tool, refer to "Chapter 9 - The File System Editor".

The Card Simulator Console

This proxy can monitor all the card simulators working at the time. The console window looks like below:

Figure 10 - Card Simulator Console


Gemalto Card Simulator Console
NFC with Proprietary SCWS:1552
NFC with Proprietary SCWS:1751

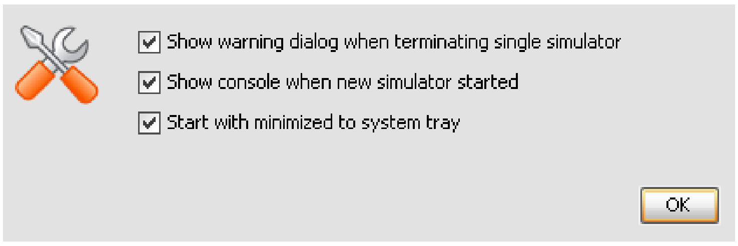
Trace Not Available.

Item	Value
Name	NFC with Proprietary SCWS
Port	1552
Started Time	Wed Jul 30 09:48:06 CST ...
Status	Card Emulation mode

On the left the tabs list all card types that is running; on the right you can see some trace information.

1 Clicking on the button will pop up the configuration dialogue. The dialogue will let you choose the parameter as you want. Click **OK** to save your changes.

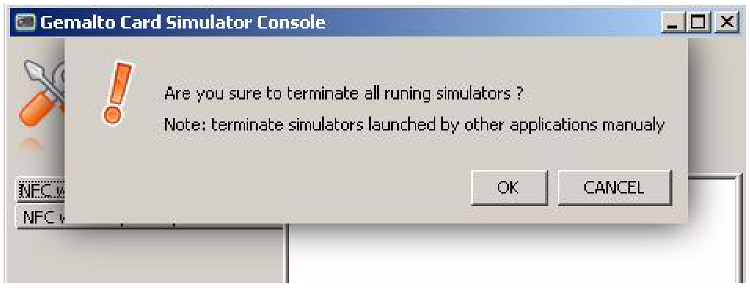
Figure 11 - Card Simulator Console - Configuration Dialogue



- [✓] Show warning dialog when terminating single simulator
- [✓] Show console when new simulator started
- [✓] Start with minimized to system tray

2 The button is used to terminate running card simulator(s). On clicking on this button, a warning message appears, asking for your confirmation.

Figure 12 - Card Simulator Console - Confirm Terminating



Are you sure to terminate all runing simulators ?

Note: terminate simulators launched by other applications manualy

Object Journalism Feature

The Object Journalism Feature is a data updating process. This process aim to ensure the flash memory is used in a uniform fashion, hence, to have a better endurance on the memory.

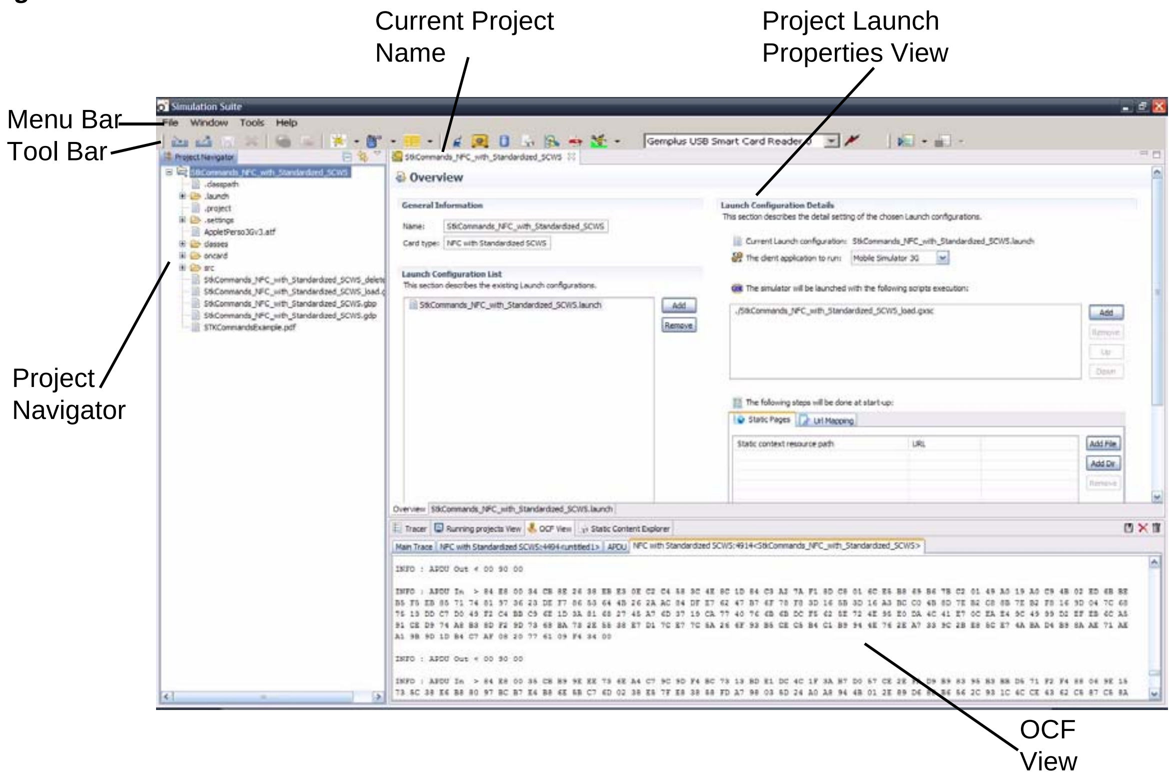
This process can be done by simply launching the Application Manager. To know more about how to activate this mechanism, please refer to “Chapter 7 - Using the Application Manager” for more details.

Getting Started with Simulation Suite V2.6

Simulation Suite V2.6 Main Screen

The figure below shows the main screen of the Simulation Suite V2.6.

Figure 13 - Simulation Suite V2.6 Main Screen



StkCommands_NFC_with_Standardized_SCWS
- .classpath
- .launch
- .project
- .settings
- AppletPerso3Gv3.atf
- classes
- oncard
- src
- StkCommands_NFC_with_Standardized_SCWS_delete
- StkCommands_NFC_with_Standardized_SCWS_load.g
- StkCommands_NFC_with_Standardized_SCWS.gbp
- StkCommands_NFC_with_Standardized_SCWS.gdp
- STKCommandsExample.pdf

StkCommands_NFC_with_Standardized_SCWS

General Information

Name: StkCommands_NFC_with_Standardized_SCWS

Card type: NFC with Standardized SCWS

Launch Configuration List

This section describes the existing Launch configurations.

StkCommands_NFC_with_Standardized_SCWS.launch

Launch Configuration Details

This section describes the detail setting of the chosen Launch configurations.

Current Launch configuration: StkCommands_NFC_with_Standardized_SCWS.launch

The client application to run: Mobile Simulator 3G

The simulator will be launched with the following scripts execution:

./StkCommands_NFC_with_Standardized_SCWS_load.gxsc

The following steps will be done at start-up:

Static context resource path	URL	

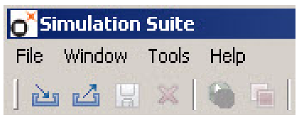
Overview | StkCommands_NFC_with_Standardized_SCWS.launch

```
INFO : APDU Out < 00 90 00

INFO : APDU In  > 84 E8 00 34 CB 8E 26 38 EB E3 0E C2 C4 58 9C 4E 8C 1D 84 C3 A2 7A F1 8D C8 01 6C E6 B8 69 B6 7B C2 01 49 A0 19 A0 C9 4B 02 ED 6B BE
B5 F6 EB 85 71 74 81 97 36 23 DE F7 86 53 64 4B 26 2A AC 84 DF E7 62 47 B7 6F 78 F8 3D 16 5B 3D 16 A3 BC C0 4B 8D 7E B2 C8 8B 7E B2 F8 16 9D 04 7C 68
75 13 DD C7 D0 49 F2 C4 BB C9 6E 1D 3A 81 68 27 45 A7 6D 37 19 CA 77 40 76 6B 6B DC F6 62 8E 72 4E 96 E0 DA 4C 41 E7 0C EA E4 9C 49 99 D2 EF EB 6C A5
91 CE D9 74 A8 B3 8D F2 9D 73 69 BA 73 2E 55 38 E7 D1 7C E7 7C 6A 26 6F 93 B5 CE C5 B4 C1 B9 94 4E 76 2E A7 33 9C 2B E8 6C E7 4A BA D4 B9 8A AE 71 AE
A1 9B 9D 1D B4 C7 AF 08 20 77 61 09 F4 34 00

INFO : APDU Out < 00 90 00

INFO : APDU In  > 84 E8 00 36 CB B9 9E EE 73 6E A4 C7 9C 9D F4 8C 73 13 BD E1 DC 4C 1F 3A B7 D0 57 CE 2E [illegible] D9 B9 83 95 B3 BB D6 71 F2 F4 88 06 9E 15
73 6C 38 E6 B8 80 97 BC B7 E4 B8 6E 6B C7 6D 02 38 E6 7F E8 38 68 FD A7 98 03 6D 24 A0 A8 94 4B 01 2E 89 D6 [illegible] B6 56 2C 93 1C 6C CE 63 62 C6 87 C6 8A
```

Menu Bar

As the following figure shows, the Simulation Suite V2.6 consists three Menus: **File**, **Tools** and **Help**.

Figure 14 - Simulation Suite Menu Bar



File Menu

This menu contains accesses to functionalities mostly related to the project management. It has four sub selection items.

- Import: Clicking this menu will display the Project Import Wizard main page.
- Export: Clicking this menu will display the Project Export Wizard main page.
- Save: Save changes to currently opened items.
- Exit: Click to close the current window. You will be prompted to save or discard any unsaved modifications. Click Yes to save or click No to discard any changes.

Window Menu

The menu offers accesses to the preferences option provided in the Simulation Suite.

Tools Menu

The menu offers accesses to all capabilities the Simulation Suite may need in a test. Tools included in menu are referred to in details in “Chapter 3 - Simulation Suite V2.6 Components”.

Help Menu

This menu contains product license and version information, as well as online help.

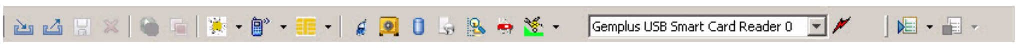
At any time, you can click **Help > View License** to view detailed license status at component level, and start to require or import a license.

About how to request a new license or use an existing one, please refer to “Apply and Import A License for Simulation Suite V2.6” on page 15.

Tool Bar

The tool bar provides quick accesses to commands that can be found in menu bar. The following figure shows all shortcuts the tool bar contains.

Figure 15 - Simulation Suite Tool Bar



To see function of an icon in the tool bar, put the mouse pointer over it for a second and the tagged description will display.

Project Navigator

This left-hand panel is also known as project explorer. It is the viewer for all project imported under workspace.

Java Card Project Import and Export

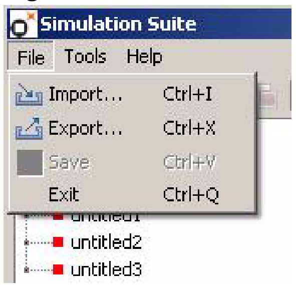
Import and Export Menu Access

Simulation Suite V2.6 provides ability to import external Developer Suite projects into Simulation Suite V2.6 and then export them with last saved Developer Suite Project and Simulation Suite tools properties.

To start an Import/Export wizard, you can do either of the following.

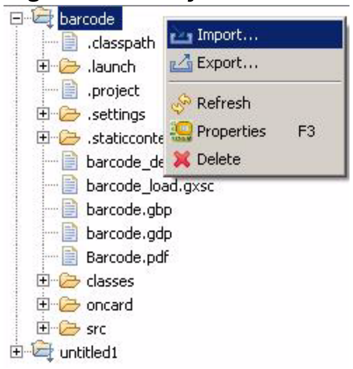
- Click the or icon in the tool bar for a direct access.
- As the following figure shows, click **File** Eclipse menu and select **Import**/**Export**.

Figure 16 - File Menu - Import & Export



- On the **Package** explorer, select the blank area and right-click to display the context sensitive menu. On the contextual menu displayed as “Figure 17”, select **Import**/**Export**.

Figure 17 - Project Contextual Menu - Import & Export

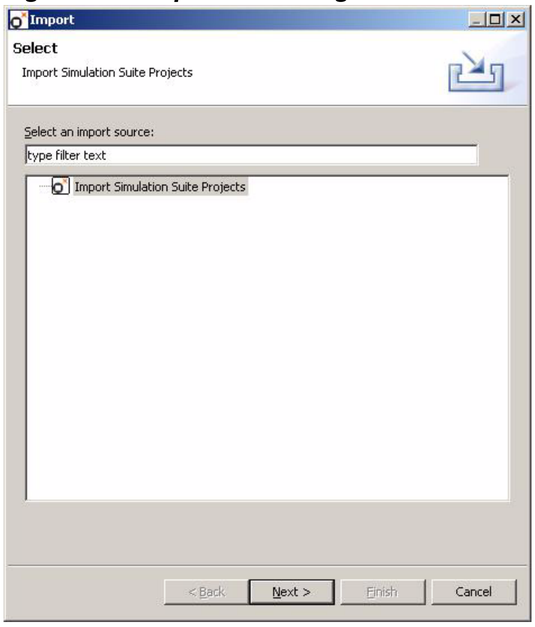


Importing Simulation Suite Projects

To import Simulation Suite Projects, you need to following these steps:

1 Click the Import icon in tool bar or select **File > Import** from the menu bar. This start the Project Import Wizard main window.

Figure 18 - Import Main Page


Import

Select

Import Simulation Suite Projects

Select an import source:

type filter text

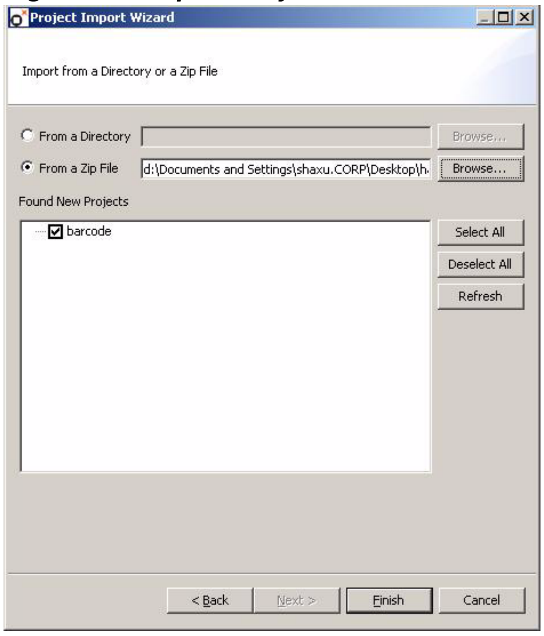
Import Simulation Suite Projects

2 Select **Import Simulation Suite Projects** and click **Next** to continue. This displays steps 1 of the **Import Projects** Wizard.

3 On the window displayed, check **From a Directory** radio button and browse to locate an the project folder to import; or select **From a Zip File** and browse to find an archive file.

4 Projects in the specified archived file or under the specified directory will be automatically detected and displayed in the **Projects** field once they are found. All the detected projects are selected by default as shown in the figure below:

Figure 19 - Import Projects Found


Project Import Wizard

Import from a Directory or a Zip File

From a Directory

[✓ From a Zip File] d:\Documents and Settings\shaxu.CORP\Desktop\h

Found New Projects

☑ barcode

5 Select the projects you want to include. Click **Finish** to complete import.

Exporting Simulation Suite Projects

To export Simulation Suite projects, you need to following these steps:

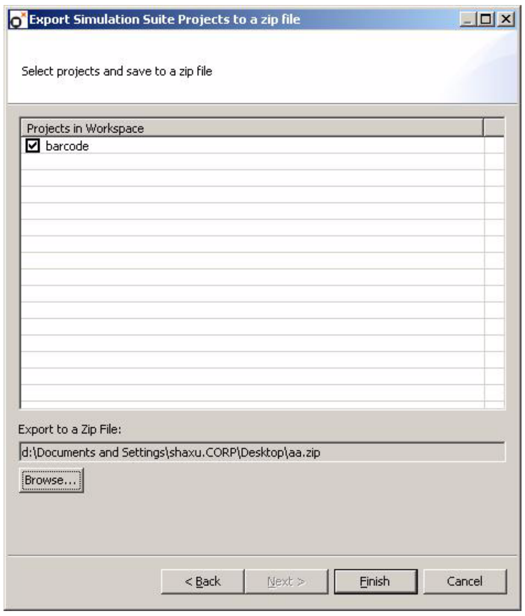
1 Click the Export icon in tool bar or select **File > Export** from the menu bar. This start the Project Export Wizard main window.

2 Select **Export as a ZIP file** radio button or select **Export to a folder**.

3 Click **Next** to proceed. This displays steps 1 of the **Export Projects** Wizard.

4 Click **Next** to display step 1 of **Simulation Suite Export Wizard**.

Figure 20* - Simulation Suite Export Wizard - *Step 1


Export Simulation Suite Projects to a zip file

Select projects and save to a zip file

Projects in Workspace
☑ barcode

Export to a Zip File:
d:\Documents and Settings\shaxu.CORP\Desktop\aa.zip

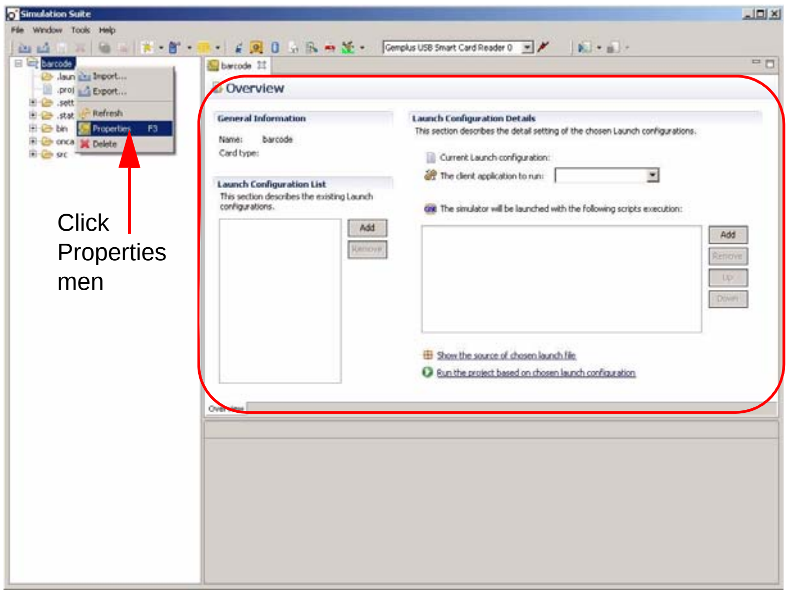
5 On the above window, select the project you want to export and click **Browse** to specify a location of the folder or .zip file for the project to export.

6 Click **Finish** to complete export.

Configure Project Launch Settings

After completion of a Simulation Suite Project Wizard, the imported project is displayed in the Project Navigator. You can right-click to display the context sensitive menu. From the menu, click **Properties** to open the project launch definition in the Project Properties Editor, as below:

Figure 21 - Project Properties Editor


Simulation Suite
File Window Tools Help
Gemplus USB Smart Card Reader 0
barcode
Import...
Export...
Refresh
Properties F3
Delete
Click
Properties
men
Overview
General Information
Name: barcode
Card type:
Launch Configuration List
This section describes the existing Launch configurations.
Add
Remove
Launch Configuration Details
This section describes the detail setting of the chosen Launch configurations.
Current Launch configuration:
The client application to run:
The simulator will be launched with the following scripts execution:
Add
Remove
Up
Down
Show the source of chosen launch file
Run the project based on chosen launch configuration
Overview

On the page displayed, shortcuts are available to modify project launching configuration.

- General Information: Read only information on the project name and project card type.
- Launch Configuration List: Project launch configuration files will be listed here. This should be a .launch file under the **.launch** folder by default. Click **Add** button to create a new file and define its details in **Launch Configuration Details** area.
- Launch Configuration Details: Specify the client application and card loading script for the currently selected launch file.
  - The client application to run:

    Choose a client application to test your project with from the following options:

    Simulation Chain 2G

    Simulation Chain 3G

    Mobile Simulator GSM

    Mobile Simulator CDMA

    Mobile Simulator 3G

    JCard Manager

    Static Content Explorer
  - The simulator will be launched with the following scripts execution: Select a `.gxsc` file which specify loading commands during simulator launching.
  - Show the source of chosen launch file: For advanced user, click this link to display the selected launch file in editor and revise the source file manually.

Run A Project

At once you finish configuring the project launch configuration, click 💾 icon on the tool bar to save modification. Then click **Run the Project based on chosen launch configuration** shortcut to start running project.

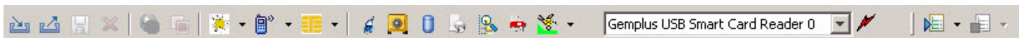
The Static Content Explorer

In this chapter, you will learn how to manage oncard static contents with the Static Content Explorer.

Menu to Access

From the Simulation Suite tool bar, click the icon. This will display the Static Content Explorer. Normally, the Static Content Explorer is automatically launched by the launching configuration where it is specified as the client application. So it's a rare case that we need to start this tool manually.

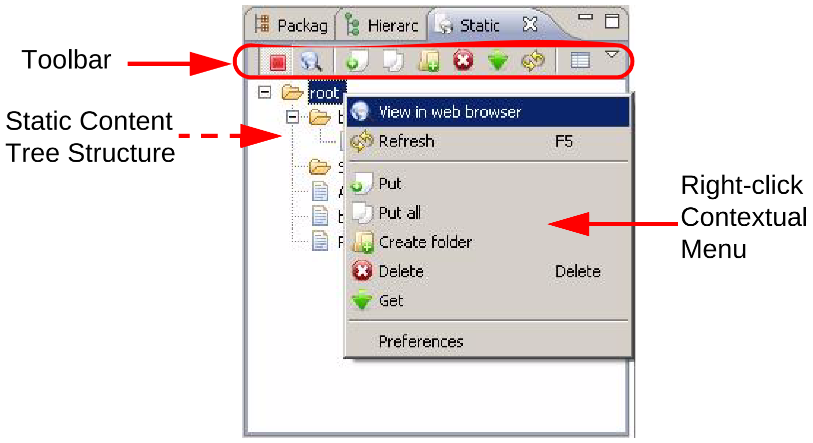
Figure 22 - Static Content Menu



Static Content Explorer View

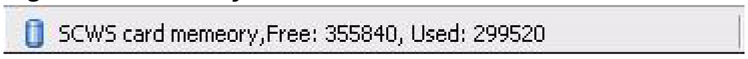
This displays the Static Content Explorer view in Eclipse workbench.

Figure 23 - Static Content Manager View



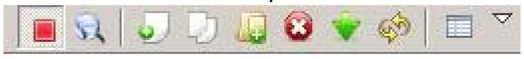
The static contents are automatically started when the Static Content Explorer is opened. The SCWS card memory status are present on the status bar (at the button of Eclipse workbench), as below:

Figure 24 - Memory status bar


SCWS card memeory,Free: 355840, Used: 299520

Static Content Explorer Toolbar

The toolbar is at the top of the Static Content Explorer.



The following contents introduce each button in the toolbar.

Add file(s) as static contents to SCWS card

The **Put** button is for recursively adding files into the SCWS card.

1 In the static content tree structure, select a folder (root included) to add the imported files.

2 Click the **Put** button to display the file explorer, where you can choose a single file or a batch of files from your PC local disk.

Tip: Press **Ctrl** and click the to select multiple files in the explorer window.

---

3 Click **Open** to add the selected file(s) to the SCWS card.

4 After adding file(s) finished, the contents structure and memory status are refreshed automatically.

Add a folder (files included) as static contents to SCWS card

The **Put All** button can recursively add a whole directory into an SCWS card while keep the same file structure with the local disk.

1 In the static content tree structure, select a folder (root included) to add the imported files.

1 Click the **Put All** button to display the file explorer.

2 On the **Browse for Folder** window displayed, browse the file system to specify a directory on your PC local disk.

3 After selecting a directory to add, click **Ok** to add files.

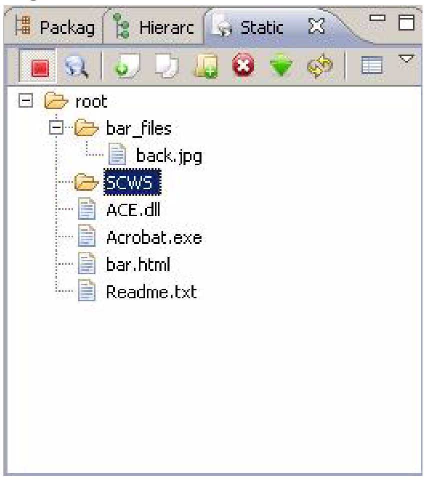
4 The contents structure and memory status are refreshed automatically.

Create an empty folder in static contents tree structure

The **Create Folder** button allows to create an empty folder under a specific node of the static contents tree structure. To add a new folder:

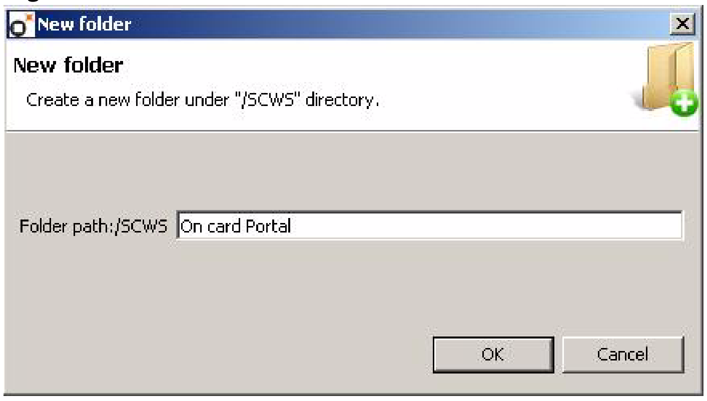
1 In the oncard static contents tree structure, click to select a parent folder for the folder to be created, e.g. **SCWS** as highlighted in the figure below:

Figure 25 - Create New Folder - Select A Parent Folder



2 Click the **Create Folder** button. This display a **New Folder** page as below:

Figure 26 - Create New Folder - Enter Path



New folder

Create a new folder under "/SCWS" directory.

Folder path:/SCWS On card Portal

3 On the window above, the select folder's path is given. The new folder will be added into that directory.

In the text field, enter a name for new folder.

Note: When you name the new folder, notice that space is allowed which a backslash (\) is forbidden.

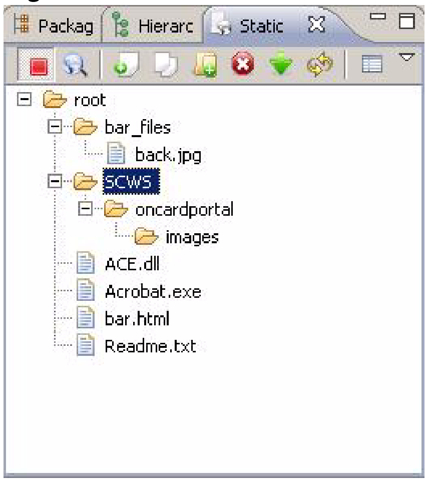
You can also use a slash (/) to including a sub folder, e.g. "On card port/images" (double quotation marks excluded).

4 Click **Ok** to add folder(s). The contents structure and memory status are refreshed automatically.

5 If you can't see update in the tree structure, click **Refresh** button or press **F5** key to refresh the display.

6 The new created folder(s) appears in the static content tree structure, as the figure shown below:

Figure 27 - Create New Folder - Content Structure Updated



Delete a file or folder

The **Delete** allows you to remove unwanted files or folders from the SCWS card.

1 In the static content tree structure, select a single file or folder to delete. Like the case for adding files, you can also choose a batch of files to delete.

Warning: To delete a folder will recursively delete all files and sub-folders under it's directory.

---

2 Click **Delete** button.

3 You will be prompted for confirmation on continuing the operation.

4 Click **Yes** to confirm or click **Cancel** to quit.

5 The contents structure and memory status are refreshed automatically.

Reload the static file structure in SCWS card

Click the **Refresh** button refresh your static content tree structure. This will:

- Refresh the root directory if no file or folder is selected.
- Refresh the corresponding directory selected.

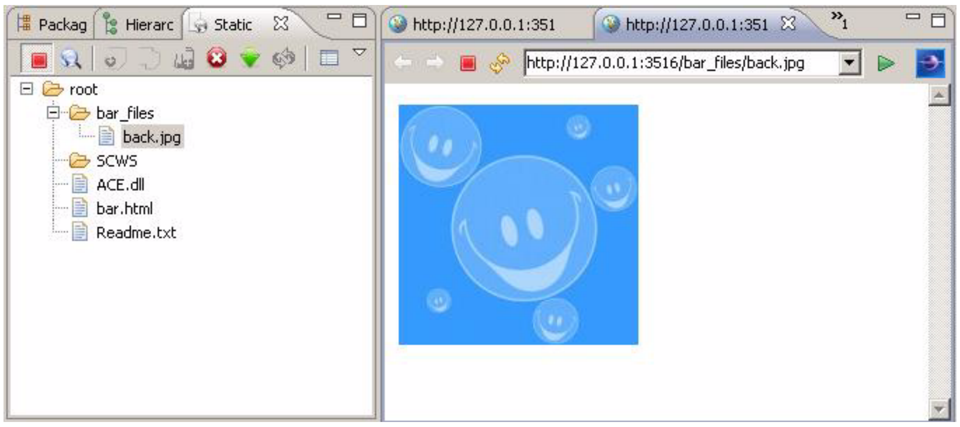
Launch the system web browser in Eclipse Editor view panel

The **View in web browser** button allows to open the system web browser in Eclipse Editor view panel to display a selected file or folder. This button is disabled if no file or folder is selected in the static content structure.

1 In the static content tree structure, select a single file or folder to display.

2 Click the **View in web browser** button to launch the web browser. The selected file or folder is opened in the web browser. As shown in the following figure:

Figure 28 - Launch Browser



3 As in “Figure 28”, the selected image file `back.jpg` is displayed in the web browser on the right.

Download a file or folder from the SCWS card to local disk

The **Get** button allows to export files or folders selected in SCWS card into local system. As the case in adding file, you can select one single file or a batch of files at a time by press **Ctrl** key and click on files.

This button is disabled if no file or folder is selected in the static content tree structure.

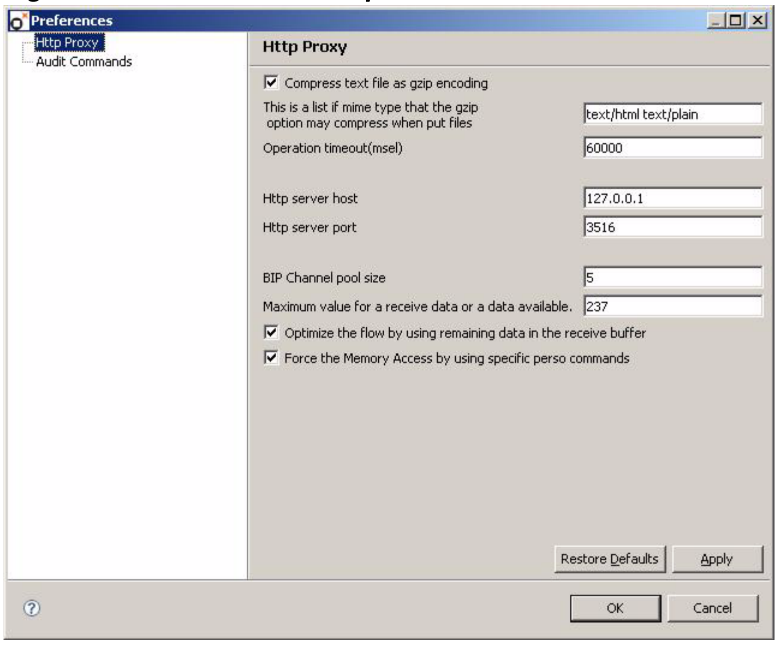
If a file with the same name already exists on the local disk, a window prompts to you, asking whether to overwrite and existing copy or not.

Set the properties of Static Content Explorer

The **Preference** button enables you to modify Static Content Explorer settings. To change preference configuration:

1 Click **Preference** button. This displays the **Preference** page.

Figure 29 - Static Content Explorer Preference



Http Proxy

☑ Compress text file as gzip encoding

This is a list if mime type that the gzip option may compress when put files	text/html text/plain
Operation timeout(msel)	60000
Http server host	127.0.0.1
Http server port	3516
BIP Channel pool size	5
Maximum value for a receive data or a data available.	237

☑ Optimize the flow by using remaining data in the receive buffer

☑ Force the Memory Access by using specific perso commands

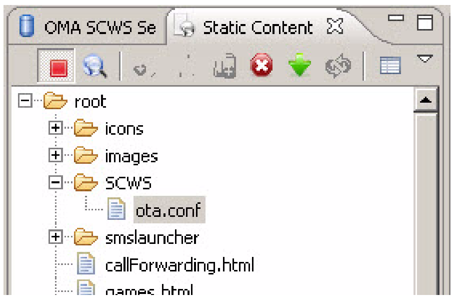
2 On the **HTTP Proxy** screen, define the following parameters or use the default values:

- **Compress text file as gzip encoding:** if gziped the file when put text file, there is a list if MIME type that the gzip option may compress when put files. Multiple types are separated by space. For example, text/html text/plain.
- **Http server Address:** The ip address of the http server, default value is 127.0.0.1.
- **Http server Port:** This is a http service port, default value is 3516.
- **Maximum receive length:** Maximum value for a receive data or a data available. Shall never exceed 237 (APDU limit), default value is 237.
- **Chained receive:** if true try to optimize the flow by using remaining data length in the receive buffer, else perform a data available between each receive. The default value is true.
- **timeout:** use for receive and emit data. On the user point of view, it looks like Http request time out. Default value is 60000 (1 min).
- **BIP Channel size:** the size of BIP channel pool, default value is 5. It can open multiple BIP channels at the same time.
- Force the Memory Access by using specific perso commands, and temporarily disable the SCWS security (for the current ATR session) if the current card is in secure mode.

After modification, you need to re-start the Static Contents Explorer to take effect.

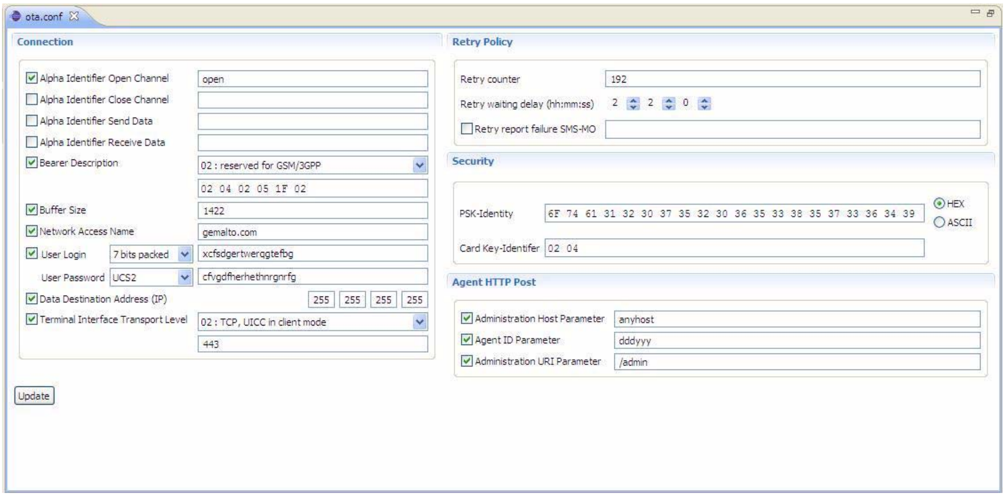
OTA Configuration Interpreter and Editor

Figure 30 - Static Content Explorer Tree Structure - SCWS > ota.conf



In the oncard static contents tree structure as shown above, click to select **ota.conf** file from the SCWS parent folder, this will open the **OTA Configuration Editor** window (see “Figure 31”) for user to make the selection and modification of the parameters. After any selection and modification, user can click the **Update** button to save the changes on the card.

Figure 31 - OTA Configuration Editor Window



ota.conf

Connection

✓ Alpha Identifier Open Channel		open
☐ Alpha Identifier Close Channel		
☐ Alpha Identifier Send Data		
☐ Alpha Identifier Receive Data		
✓ Bearer Description		02 : reserved for GSM/3GPP
		02 04 02 05 1F 02
✓ Buffer Size		1422
✓ Network Access Name		gemalto.com
✓ User Login	7 bits packed	xcfsdgertwergqtefbg
User Password	UCS2	cfvgdfherhethnrgnrfg
✓ Data Destination Address (IP)		255 255 255 255
✓ Terminal Interface Transport Level		02 : TCP, UICC in client mode
		443

Update

Retry Policy

Retry counter	192
Retry waiting delay (hh:mm:ss)	2 2 0
☐ Retry report failure SMS-MO	

Security

PSK-Identity	6F 74 61 31 32 30 37 35 32 30 36 35 33 38 35 37 33 36 34 39
Card Key-Identifer	02 04

[✓ HEX] ASCII

Agent HTTP Post

✓ Administration Host Parameter	anyhost
✓ Agent ID Parameter	dddyyy
✓ Administration URI Parameter	/admin

This user interface has four main sections:

- Connection Parameters
- Retry Policy Parameters
- Security Parameters
- Agent Http Post Parameters

Connection Parameters

The parameters in this section allow you to establish a point-to-point TCP connection between the Smart Card Web Server and the service provider.

Check on the box of each parameter to activate the connection between the Smart Card Web Server and the server.

Alpha Identifier Open Channel

Contains the name (alpha-tag) associated with the BIP **Open Channel** proactive command.

Alpha Identifier Close Channel

Contains the specified Gemalto proprietary name (alpha-tag) to use during **Close Channel** proactive command.

Alpha Identifier Send Data

Contains the name (alpha-tag) associated with the **Send Data** proactive command.

Alpha Identifier Receive Data

Contains the name (alpha-tag) associated with the **Receive Data** proactive command.

Bearer Description

Contains the bearer information associated with the BIP **Open Channel** proactive command.

Buffer Size

Contains the maximum buffer size associated with the BIP **Open Channel** proactive command.

Network Access Name

Contains the name (alpha-tag) used to identify the gateway entity which provides the interworking with an external network associated with the BIP **Open Channel** proactive command.

User Login

Contains the login access (text string) to the current application's files associated with the BIP **Open Channel** proactive command. The encoding format for this is to chosen from the drop-down list.

User Password

Contains the User Password associated with the BIP **Open Channel** proactive command. The encoding format for this is to chosen from the drop-down list.

Data destination Address (IP)

Contains the destination address associated with the BIP **Open Channel** proactive command.

Terminal Interface Transport Level

Contains the transport level tag associated with the BIP **Open Channel** command. Select IPv4 or IPv6 to be used as the network layer protocol.

Retry Policy Parameters

The parameters in this section are used to reconnect the terminated session due to administrative failures such as network coverage failure or when the server is not responding. It contains the retry counter (the initial value set by the retry policy), the retry waiting delay (minimum duration between two retries) and the retry failure SMS-MO (failure report sent to the server when request is aborted).

Check on the box of each parameter to activate the connection between the Smart Card Web Server and the server.

Retry counter

Indicates the number of attempts before reconnection is aborted.

Retry waiting delay (hh:mm:ss)

Indicates the waiting time between two attempts.

Retry report failure SMS-MO

Check this box to generate a report when reconnection fails. This value can be taken from the TP-Destination Address, Service Center Address or Alpha Identifier in the TLV format to indicate that there is a failure report from SMS-MO.

Security

The parameters in this section allow you to configure the Transport Layer Security (TLS) layer. It contains the PSK-Identity parameter containing the keys chosen by the SCWS which the server needs to authenticate, and the card key-identifier parameter that is used by the card to check the key value.

Check on the box of each parameter to enable the security between the Smart Card Web Server and the server.

PSK-Identity

Contains the PSK identity used by the remote server during the TLS session.

Card Key-Identifier

Contains the key value to be used during the TLS session.

Agent HTTP Post

This is one of the HTTP methods employed by the card to send a Post request to the SCWS.

The followings are the parameters:

Administration Host Parameter

Defines the Host header value in the Post request.

Agent ID Parameter

Defines the ICCID of the card to allow identification by the host remote server in a Post request.

Administration URI Parameter

Defines the URI value in the Post request to the host remote server.

The JCardManager

The JCardManager is a powerful tool that provides the following functions:

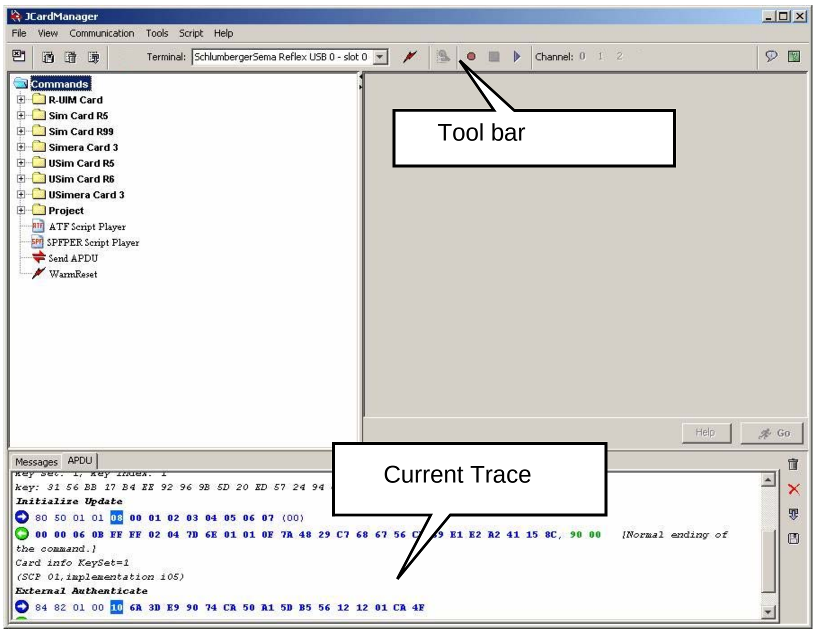
- Perform individual command APDUs on a card or a card simulator
- Record and replay script files (sequence of commands)
- Display the results of APDU exchanges with the card or simulator and save them in a trace file
- A file system editor enabling you to view the files in a card and their properties. This is described separately in “Chapter 9 - The File System Editor”.

JCardManager can be started in one of two ways:

- In Eclipse, from the **Developer Suite** menu, choose **JCardManager**.
- From the **Start** button, choose **All Programs > Gemalto > Wireless Solutions > JCardManager**.

Both these actions open the initial **JCardManager** window as shown in the following figure:

Figure 32 - The Initial JCardManager Window



```
Key Set: 1, Key Index: 1
key: 31 56 BB 17 B4 EE 92 96 9B 5D 20 ED 57 24 94
Initialize Update
80 50 01 01 08 00 01 02 03 04 05 06 07 (00)
00 00 06 0B FF FF 02 04 7D 6E 01 01 0F 7A 48 29 C7 68 67 56 C[illegible]9 E1 E2 A2 41 15 8C, 90 00   [Normal ending of
the command.]
Card info KeySet=1
(SCP 01,implementation i05)
External Authenticate
84 82 01 00 10 6A 3D E9 90 74 CA 50 A1 5D B5 56 12 12 01 CA 4F
```

For detailed help about JCardManager, click the **JCardManager Help** button . This section provides an outline of the main operations that you will want to perform.

For detailed help about a particular command, select the command in the command list and click **Help** at the bottom right of the window.

To execute a command:

1. Expand the tree structure in the left pane and select the command. The parameters for the command display in the right pane.
2. Complete the parameters in the right pane (these could include option buttons, text fields, check boxes and so on and vary according to the command chosen).
3. Click **Go**.

Logical Channels

To select a logical channel, click the corresponding button in the multichannel bar. APDUs will then be sent to this logical channel.

Script Files

Script files contain a sequence of command APDUs. You can make your own script file by recording a sequence of commands and play back the script file.

There are two main types of script file:

- `.atf` files, which store the data values directly.
- `.xml` and `.gxsc`, which store the parameters and rebuild the APDUs, for example they recalculate random numbers for you.

In JCardManager you can use the record and play tools described below to record and play both types of file. However, JCardManager also contains specific script players, the ATF Script player and the XML Script player (which can also play `.gxsc` scripts).

To record a script file:

1. Start recording by clicking ● in the toolbar or by choosing **Script** > **Record** or pressing the <F5> key.
2. In the **Record Script. Output File Selection** window, browse to the directory where you want to save the script file and enter the file name (with a `.gxsc` or `.xml` suffix).
3. Click **Save**.
4. Perform the sequence of commands that you want to record in your script.
5. After the last command, stop recording by clicking ■ in the toolbar or by choosing **Script** > **Stop** or pressing the <F10> key.

Note: During recording you can pause recording by pressing ⏸ or by choosing **Script** > **Hold** or pressing F9. This stops commands being written to the script file. To continue writing commands to the script file press ● or choose **Script** > **Record** or pressing F5.

To play a script file (general):

1. Start playing by clicking ▶ in the toolbar or by choosing **Script** > **Play** or pressing F12.
2. In the **Play Script. File Selection** window, browse to the directory that contains the script file you want to play and select the file.

3 Click **Open**.

To play an .atf script using the ATF Script Player:

1 Select **ATF Script Player** in the left pane.

2 In **ATF file**, either select the `.atf` file from the list (the list stores all the previously selected `.atf` files) or navigate to an `.atf` file.

3 Use the ATF Script Command buttons to:

- Rename a command
- Change the order of the commands in the list
- Delete one or more commands from the list
- Save the `.atf` file under the same or a different name.

4 Click **Go** to play the script. The results of the script execution are displayed in the **Message** area of the window.

Note: For more detailed information about the ATF Script Player, click **Help**.

To play an .xml or .gxsc script using the XML Script Player:

1 Expand the folder in the left pane corresponding to your type of card, for example USim Card R5, and select **XML Script Player**.

2 In **GXSC(XML) file**, either select the file from the list (the list stores all the previously selected `.gxsc` and `.xml` files) or click the browse button and navigate to a file.

Note: You can also edit XML files graphically using the Script Editor tool.

3 Use the **Command List** buttons to:

- Rename a command
- Change the order of the commands in the list
- Delete one or more commands from the list
- Save the `.gxsc` or `.atf` file under the same or a different name.

4 Click **Go** to play the script. The results of the script execution are displayed in the **Message** area of the window.

Note: For more detailed information about the XML Script Player, click **Help**.

Traces

The lower pane in the JCardManager displays a log file known as a trace. You can make this trace pane disappear and reappear (known as toggling), clear its contents or save the contents to a file.

To make the trace pane disappear or reappear:

Click in the toolbar or choose **View** > **Trace** > **SwitchTrace Action** or press Alt+X.

To clear the trace pane:

Click in the toolbar or choose **View** > **Trace** > **CleanTrace Action** or press Alt+E.

To save the trace in a file:

Click in the toolbar or choose **View** > **Trace** > **SaveTrace Action** or press Alt+S.

Using the Application Manager

This chapter introduces the Application Manager tool supplied with Simulation Suite V2.6. The Application Manager is used to download packages onto real cards and install and delete applets or applet instances.

Introduction

The main features of the Application Manager are that it:

- Provides an easy to use “application repository” in which to store information about the applications that are ready to be loaded onto cards. You can create, copy, import, export, and delete applications and modify their properties.
- Supports downloading applications to real cards.
- Supports the following loading modes:
  - “Over-the-air” (OTA) mode, which uses the 3GPP TS 23.048 Short Message Service (SMS) facility to remotely download applications. The Application Manager generates the appropriate envelope commands and parameters and sends the application to the target card encapsulated within one or more SMS point-to-point data download messages.
  - “Input/Output” (I/O) mode, with which the Application Manager sends GlobalPlatform commands directly to the Card Manager applet on the target card.
  - Bearer Independent Protocol (BIP) mode, with which the Application Manager sends BIP commands in order to perform applet management.
- Uses *card profiles* to configure all the parameters necessary to send applications to specific card types in I/O or OTA mode. Each card profile is based on a template. A preconfigured and fully tested template is supplied for each of the sample card types delivered with the product you have purchased, for example, the GemXplore 3G V3. You can create, copy, and delete card profiles and modify their properties as necessary.
- Provides flexibility in terms of the tasks you can perform. For example, you can load packages that contain only Java library classes referenced by other applets, load packages and install several different applets from the package, or create multiple instances of an applet that has already been loaded onto the card. You can also choose to perform separate load and install operations. For example, you load a package initially, then install the applets it contains later.

- Allows you to debug the load and install process by means of an easy to understand trace window.

Overview of Defining and Running an Application

To define and run an application:

1. Start the Application Manager.
2. Configure the terminal profile settings.
3. Select the terminal containing the target card.
4. Create, configure, or select the application to load into the target.
5. Select the loading mode.
6. Create, configure, or select a suitable target card profile for the card.
7. Select the action to perform on the application.
8. Execute the selected action.
9. Optionally, analyze the results in the **Trace** panel.

For detailed instructions on performing these tasks, start the Application Manager then click the Help button in the top right-hand corner of the Application Manager window that appears.

Starting the Application Manager

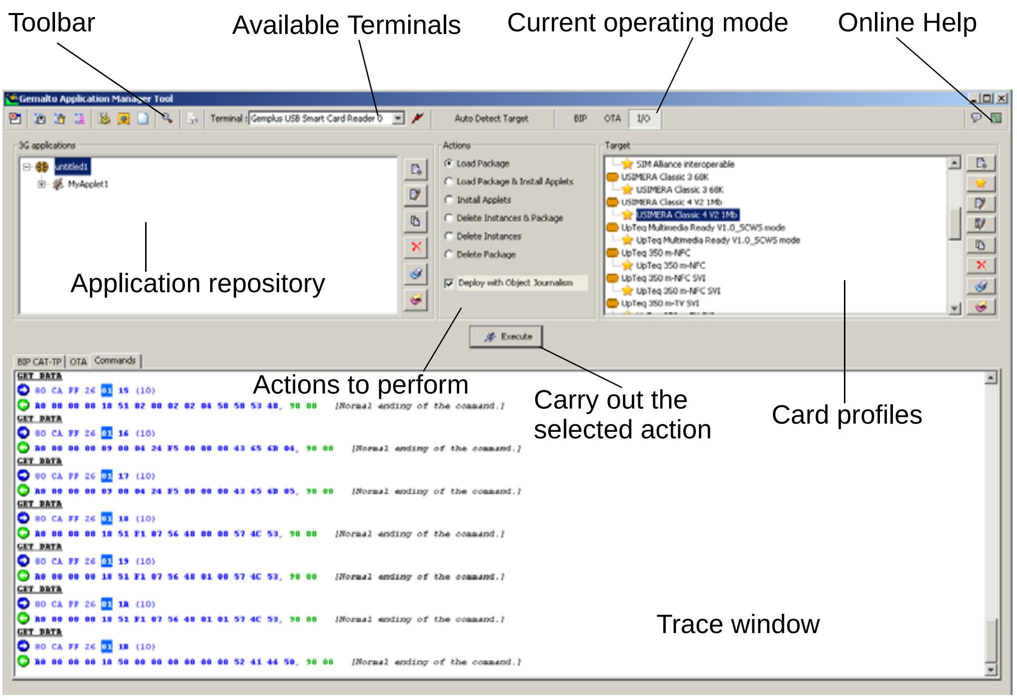
Application Manager can be started from any of the followings*:*

- Start > All Programs > Gemalto > Wireless Solutions > Application Manager 2G or Application Manager 3G.
- Simulation Suite
  1. Start Simulation Suite
  2. Choose Tools > Application Manager

The Application Manager Window

When first started, the main window of the Application Manager is displayed, as shown in “Figure 33”:

Figure 33 - The Application Manager Main Window



```
GET DATA
80 CA FF 26 01 15 (10)
A0 00 00 00 18 51 02 00 02 02 04 50 50 53 48, 90 00   [Normal ending of the command.]
GET DATA
80 CA FF 26 01 16 (10)
A0 00 00 00 09 00 04 24 F5 00 00 00 43 65 6D 04, 90 00   [Normal ending of the command.]
GET DATA
80 CA FF 26 01 17 (10)
A0 00 00 00 09 00 04 24 F5 00 00 00 43 65 6D 05, 90 00   [Normal ending of the command.]
GET DATA
80 CA FF 26 01 18 (10)
A0 00 00 00 18 51 F1 07 56 48 00 00 57 4C 53, 90 00   [Normal ending of the command.]
GET DATA
80 CA FF 26 01 19 (10)
A0 00 00 00 18 51 F1 07 56 48 01 00 57 4C 53, 90 00   [Normal ending of the command.]
GET DATA
80 CA FF 26 01 1A (10)
A0 00 00 00 18 51 F1 07 56 48 01 01 57 4C 53, 90 00   [Normal ending of the command.]
GET DATA
80 CA FF 26 01 1B (10)
A0 00 00 00 18 50 00 00 00 00 00 00 52 41 44 50, 90 00   [Normal ending of the command.]
```

Application Manager’s Card Auto Detection

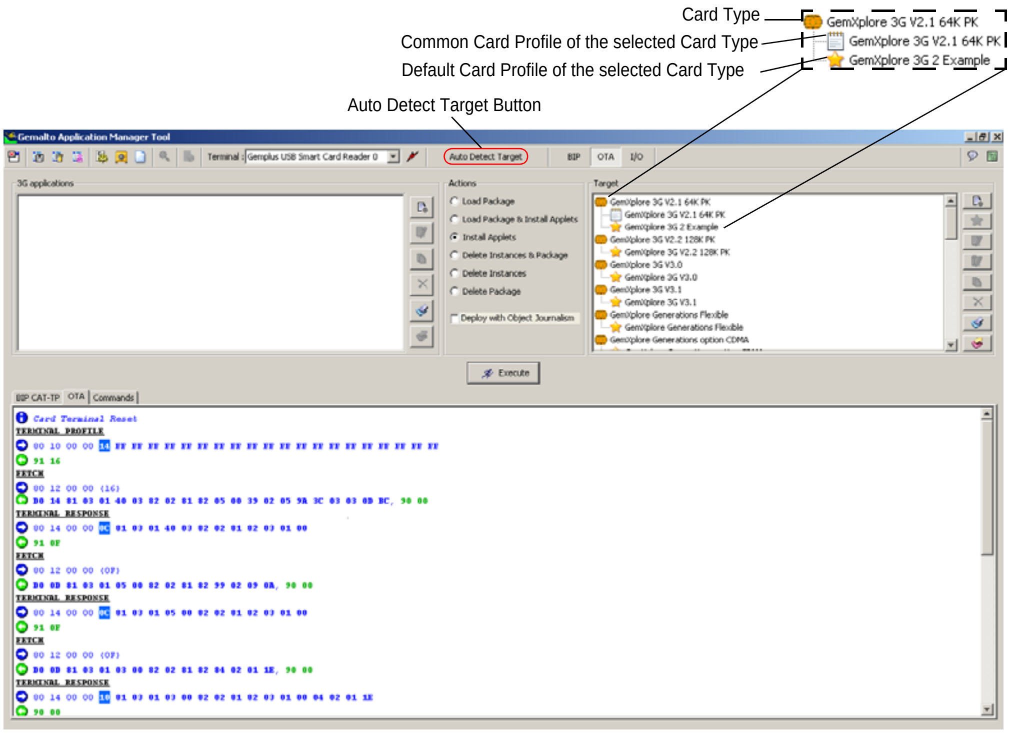
Card Profile Auto Detection is a feature provided on Application Manager. It enables user to select automatically the correct target profile for each inserted card. It will detect the card type of the card that is being inserted into the terminal and map the card type with it associated target profile in the **Target** area of the Application Manager.

The general steps will be:

1. Insert card into user’s card reader, the terminal name will be shown on the **Terminal** drop down box user interface.
2. Click on **Auto Detect Target** button to perform profile auto detection.
3. Application Manager will select the correct target profile as shown in the **Target** area automatically according to the card type being inserted.

Users can re-define their default Card Profile by selecting the common Card Profile, and then click on the ⭐ button.

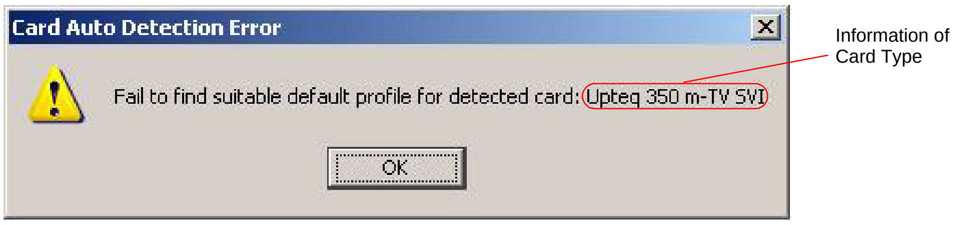
Figure 34 - Gemalto Application Manager Tool window


Card Type
GemXplore 3G V2.1 64K PK
Common Card Profile of the selected Card Type
GemXplore 3G V2.1 64K PK
Default Card Profile of the selected Card Type
GemXplore 3G 2 Example
Auto Detect Target Button
Gemalto Application Manager Tool
Terminal : Gemplus USB Smart Card Reader 0
Auto Detect Target
BIP
OTA
I/O
3G applications
Actions
Load Package
Load Package & Install Applets
Install Applets
Delete Instances & Package
Delete Instances
Delete Package
Deploy with Object Journalism
Target
GemXplore 3G V2.1 64K PK
GemXplore 3G V2.1 64K PK
GemXplore 3G 2 Example
GemXplore 3G V2.2 128K PK
GemXplore 3G V2.2 128K PK
GemXplore 3G V3.0
GemXplore 3G V3.0
GemXplore 3G V3.1
GemXplore 3G V3.1
GemXplore Generations Flexible
GemXplore Generations Flexible
GemXplore Generations option CDMA
Execute
BIP CAT-TP
OTA
Commands

```
Card Terminal Reset
TERMINAL PROFILE
00 10 00 00 14 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
91 16
FETCH
00 12 00 00 (16)
D0 14 81 03 01 40 03 82 02 81 82 05 00 39 02 05 9A 3C 03 03 0D BC, 90 00
TERMINAL RESPONSE
00 14 00 00 0C 01 03 01 40 03 02 02 01 02 03 01 00
91 0F
FETCH
00 12 00 00 (0F)
D0 0D 81 03 01 05 00 82 02 81 82 99 02 09 0A, 90 00
TERMINAL RESPONSE
00 14 00 00 0C 01 03 01 05 00 02 02 01 02 03 01 00
91 0F
FETCH
00 12 00 00 (0F)
D0 0D 81 03 01 03 00 82 02 81 82 84 02 01 1E, 90 00
TERMINAL RESPONSE
00 14 00 00 10 01 03 01 03 00 02 02 01 02 03 01 00 04 02 01 1E
90 00
```

If the card cannot be supported by the Card Auto Detection function, Card Auto Detection Error windows will be displayed. “Figure 35” is an example of the possible Card Auto Detection Error window.

Figure 35 - Example of Card Auto Detection Error window.



Fail to find suitable default profile for detected card: Upteq 350 m-TV SVI

In this case, user has to manually select/create the card profile or the corresponding card profile based on the information of the card type provided in the error message box (for example, Upteq 350 mTV SVI).

Object Journalism Feature

Object Journalism feature is a Gemalto proprietary feature. This feature must be present in the card and Object Journalism dedicated buffer must have been created with appropriate size. Gemalto exclusive Object Journalism functionality (Endurance Booster) protects the card from stressing applets / packages and warranties a higher endurance.

The Benefit of Object Journalism Feature

Flash memory has a finite number of erase-write cycles, and the number of the cycles that the flash memory can endure, will understand as the endurance cycle.
The Object Journalism Feature is using Wear Leveling process where the data updated by the applet will be distributed to a buffer (known as Journal Buffer) allocated in the card. This is to ensure the flash memory is used in a uniform fashion, in order to extend the overall lifetime of the flash memory.

Object Journalism Feature in Application Manager Tool

The Application Manager provides the tool to load / install application into a card with Object Journalism, allowing it to support high endurance in a card. This is achievable through selecting the **Deploy with Object Journalism** on the main window (see “Figure 33” for the **Gemalto Application Manager Tool** window) with the right card (the card supporting Object Journalism Feature). A high endurance applet will be handled differently by the OS, and normal applets will not be affected by these handling.

Object Journalism Feature Verification

The **Get Data** command is used to retrieve information about which applets in the card support high endurance. Two new tags have introduced for **Get Data** command to meet the verification requirement:

- Object Journalism Applications Entry List (tag = FFh25h).
  In this tag, the **Get Data** command is used to read the new entries of Object Journalism applet, and the data output contains the list of entry number in the card. Each byte is an entry number and it represents one application (it can be library, applet or instance) in the card. The entry number will be used in the **Get Data** - **Tag Object Journalism Entry AID value** to get the AID value for this application, and it is shown in “Figure 36 - List AIDs of Object Journalism Applications”.
  The returned application AID and the status word will be displayed on the trace panel in the main window (see “Figure 33” for the **Gemalto Application Manager Tool** window) to provides the verification status of the Object Journalism application.

  “Table 4 - Status Word for Get Data - Object Journalism Applications Entry List” as shown below listed all possible status to indicate the allocation status of the journal buffer.

Table 4 - Status Word for Get Data - Object Journalism Applications Entry List

Status	**Description**
9000	Normal ending of the command.
6700	P1P2 tag does not exist.
6982	Security status is not satisfied. Verify Key is required to grant the access.
6985	Condition of use is not satisfied. P1P2 is correct, but Journal Buffer is not allocated.
6CXX	P1P2 is correct and the Journal Buffer is allocated, but P3 is of wrong length. XX indicates the correct length.
6D00	Instruction is not supported
6E00	Class is not supported

- Object Journalism Applications Entry AID value (tag = FFh26h).
  In this tag, the **Get Data** command is used to return the application AID of given entry number in data input, in which the one byte entry number is taken from the Object Journalism Applications Entry List (tag = FFh25h).
  The returned application AID and the status word will be displayed on the trace panel in the main window (see “Figure 33” for the **Gemalto Application Manager Tool** window) to provides the verification status of the Object Journalism application.

  “Table 5 - Status Word for Get Data - Object Journalism Applications Entry AID Value” as shown below listed all possible status to indicate the allocation status of the journal buffer.

Table 5 - Status Word for Get Data - Object Journalism Applications Entry AID Value

Status	Description
9000	Normal ending of the command.
6700	P1P2 tag does not exist.
6982	Security status is not satisfied. Verify Key is required to grant the access.
6985	Condition of use is not satisfied. P1P2 is correct, but Journal Buffer is not allocated.
6CXX	P1P2 is correct and the Journal Buffer is allocated, but P3 is of wrong length. XX indicates the correct length.
6A88	Reference data not found. The entry number is not a valid entry.
6D00	Instruction is not supported
6E00	Class is not supported

Note: The information given in “Table 4 - Status Word for Get Data - Object Journalism Applications Entry List” and “Table 5 - Status Word for Get Data - Object Journalism Applications Entry AID Value” are correct at the time the document is generated, and is subject to change.

Deploy Application with Object Journalism

In Application Manager, user will have a choice to deploy application (load as well as install instance) in high endurance area of the card through the selection of **Deploy with Object Journalism** option.

After launching the Application Manager, user:

- selects the targeted applet in Application Manager,
- selects the correct card (the card with endurance capability) to be inserted in the reader, and select the corresponding card profile from the **Target** panel,
- selects one of the following actions,
  - Load Package
  - Load Package & Install Applets
  - Install Applets
- checks the **Deploy with Object Journalism**.

Click the **Execute** button, Application Manager will send **Get Data** APDU commands to check the capability of object journalism supported by the card. If it is supported, the tool will deploy applet in high endurance area of the card. Otherwise, Application Manager will display the error information on the Trace panel in the main window, and stop the loading / installing process.

Note: If the **Deploy with Object Journalism** is selected, selecting the **Load Package**, **Load Package & Install Applets** or **Install Applets** will mean to load and/or install application in the high endurance area. Otherwise application management according to GP standard.
However, selecting of **Delete Instances & Package**, **Delete Instances** or **Delete Package** options, will remain in normal working, regardless the **Deploy with Object Journalism** is selected or not.

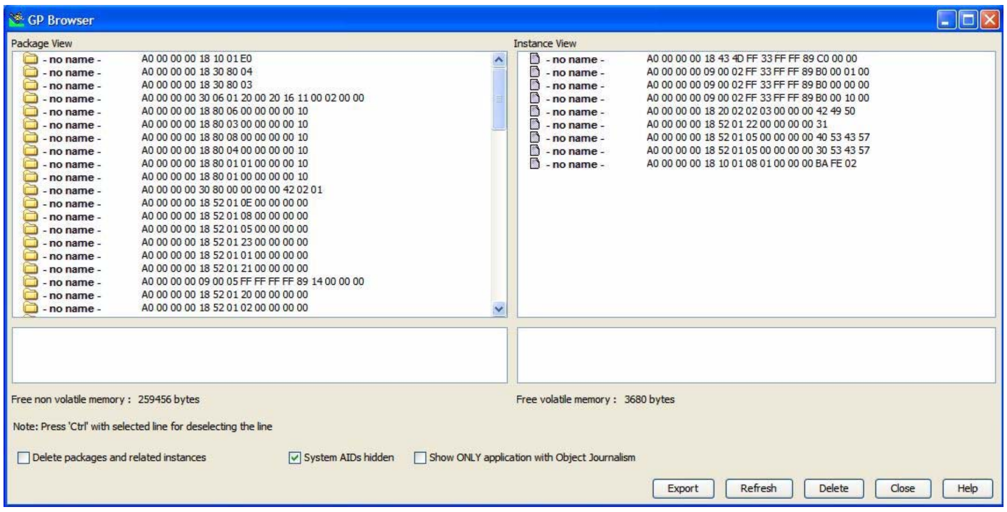
AID Browsing for Application with Object Journalism

Click on **Browse and Delete AID** button, the **GP Brower** window (see “Figure 36”) appears to allow you to view the list of Object Journalism applications available in card.

In the **GP Browser** window, check to select the **Show ONLY application with Object Journalism** checkbox, the window will list all the packages and instance (name and AID) that is deployed in high endurance area of card.

If the **Show ONLY application with Object Journalism** is unselect, the tool will refresh the package and instance AIDs to display all the packages and instances in card which following the OP/GP standard.

Figure 36 - List AIDs of Object Journalism Applications



Package View

```
- no name -    A0 00 00 00 18 10 01 E0
- no name -    A0 00 00 00 18 30 80 04
- no name -    A0 00 00 00 18 30 80 03
- no name -    A0 00 00 00 30 06 01 20 00 20 16 11 00 02 00 00
- no name -    A0 00 00 00 18 80 06 00 00 00 00 10
- no name -    A0 00 00 00 18 80 03 00 00 00 00 10
- no name -    A0 00 00 00 18 80 08 00 00 00 00 10
- no name -    A0 00 00 00 18 80 04 00 00 00 00 10
- no name -    A0 00 00 00 18 80 01 01 00 00 00 10
- no name -    A0 00 00 00 18 80 01 00 00 00 00 10
- no name -    A0 00 00 00 30 80 00 00 00 00 42 02 01
- no name -    A0 00 00 00 18 52 01 0E 00 00 00 00
- no name -    A0 00 00 00 18 52 01 08 00 00 00 00
- no name -    A0 00 00 00 18 52 01 05 00 00 00 00
- no name -    A0 00 00 00 18 52 01 23 00 00 00 00
- no name -    A0 00 00 00 18 52 01 01 00 00 00 00
- no name -    A0 00 00 00 18 52 01 21 00 00 00 00
- no name -    A0 00 00 00 09 00 05 FF FF FF FF 89 14 00 00 00
- no name -    A0 00 00 00 18 52 01 20 00 00 00 00
- no name -    A0 00 00 00 18 52 01 02 00 00 00 00
```

Instance View

```
- no name -    A0 00 00 00 18 43 4D FF 33 FF FF 89 C0 00 00
- no name -    A0 00 00 00 09 00 02 FF 33 FF FF 89 B0 00 01 00
- no name -    A0 00 00 00 09 00 02 FF 33 FF FF 89 B0 00 00 00
- no name -    A0 00 00 00 09 00 02 FF 33 FF FF 89 B0 00 10 00
- no name -    A0 00 00 00 18 20 02 02 03 00 00 00 42 49 50
- no name -    A0 00 00 00 18 52 01 22 00 00 00 00 31
- no name -    A0 00 00 00 18 52 01 05 00 00 00 00 40 53 43 57
- no name -    A0 00 00 00 18 52 01 05 00 00 00 00 30 53 43 57
- no name -    A0 00 00 00 18 10 01 08 01 00 00 00 BA FE 02
```

Free non volatile memory : 259456 bytes

Free volatile memory : 3680 bytes

Note: Press 'Ctrl' with selected line for deselecting the line

☐ Delete packages and related instances ☑ System AIDs hidden ☐ Show ONLY application with Object Journalism

OMA SCWS Server Simulator

Introduction

This chapter starts with an introduction of Smart Card Web Server (SCWS), in order to give the user a basic understanding of SCWS, before proceed to the OMA SCWS Server Simulator that integrated in the Simulation Suite V2.6 tool, in the later section.

Prerequisites

To have an understanding of this chapter, user should acquire the basic knowledge of the followings:

- Smart Card Web Server (SCWS):
  is a web server residing on the card that enables service providers to extend their multimedia services via the HyperText Transfer Protocol (HTTP).
- HyperText Transfer Protocol (HTTP):
  is a protocol which the SCWS shall implement. HTTP 1.1 standard is adopted in SCWS.
- Pre-Shared Key-Transport Layer Security (PSK-TLS):
  is a type of security used in SCWS that provides a secure and reliable transport mechanism between two communicating parties. It provides confidentiality and integrity protection for the transport used.
  HTTP over TLS protocol is adopted in SCWS.
- Open Mobile Alliance (OMA) Standard:
  is a specification that define industry-wide interoperable mechanisms for developing applications and services that are deployed over wireless communication networks. For SCWS, please refer to “*OMA-TS-Smartcard_Web_Server-V1_1-20090512-A”* specification.

The SCWS Technology

SCWS is a HTTP server implemented in a smart card, embedded in the mobile device. It allows:

- local communication between a Web browser running in the terminal and a Web Server running in the smart card.
- remote administration of the smart card web server by authorized entities (i.e. card issuer or network operator).

Applications in the card registered to the SCWS are identified by the Uniform Resource Identifiers / Uniform Resource Locators (URI/URL). You may browse the static HTML-based pages stored on the card to connect to the SCWS. The content exchange between the SCWS and the service provider is then done via the HTTP protocol defined in the OMA specifications.

The SCWS Operation Modes

Client Mode

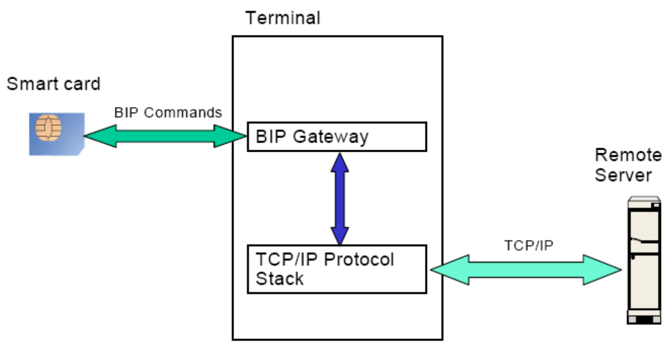
The client mode is dedicated to the Card Issuer in order to administrate the SCWS via specific administrative commands and secured accesses.

In client mode, the Card Issuer can:

- remotely administrate the SCWS by the way of OTA messages. These messages contain administrative commands and are secured via the Transport Layer Security (TLS) protocol.
- locally administrate the SCWS by the way an Application Programming Interface (API) or a Polling event called from a server applet.

The “Figure 37” below illustrates a basic functionality of SCWS in client mode.

Figure 37 - SCWS in Client Mode

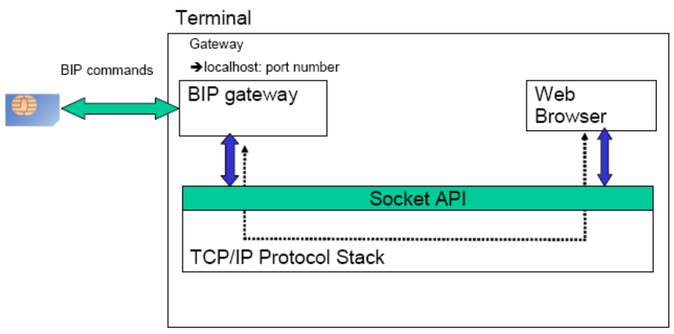


Server Mode

The server mode is initiated by the subscriber via the Web browser in the handset, for browsing the SCWS. Several channels can be opened at the same time by the subscriber, but the SCWS responds to each request, one after the others.

The “Figure 38” below illustrates a basic functionality of SCWS in server mode.

Figure 38 - SCWS in Server Mode



SCWS Administration Protocol

The SCWS administration is the ability to upload new data (e.g. xHTML pages), delete data and change configuration parameters for the SCWS. It is suitable for the exchange of a large amount of data between the administration application and the SCWS. The HTTP requests are sent using the administration protocol to SCWS for administrating the SCWS. The administration protocol enables the use of a standard web server as a remote administration server implementation.

To remotely administrate the SCWS, a card administration agent (a real HTTP 1.1 client) is in charge:

- to manage connection establishment, and
- to encapsulate and transparently transport any HTTP exchange,

between a remote administration server and the SCWS.
It also responsible of retry and reconnection management in case of communication break down.

The card administration agents can be triggered either by external events (e.g. SMS) or by internal events (internally generated by the card) and initiate a connection to a remote administration server.

SCWS Administration Commands

The HTTP requests (also known as the HTTP commands or the SCWS administration commands) are sent using the administration protocol to SCWS for administrating the SCWS, and these commands are:

- **Put** command:
  this is used to install or update a page on the SCWS.
- **Delete** command:
  this is used to delete a page or a directory from the SCWS. If the path identifies a directory, this directory and its content shall be deleted.
- **Post** command:
  this is used to send special administration commands to the SCWS. This command may also be used to send HTTP commands that pass some parameters to applications that are invoked by the SCWS. In this case the **Post** request shall include the URI / URL of the invoked application and contain the parameters in the **Post** command body.
- **Get** command:
  this is used by the admin server to read a page from the SCWS.

Administration Session Flow in Push/Pull Mode

The smart card is able to include one or several administration agents to connect to a remote administration server by opening a BIP TCP (Bearer Independent Protocol type of Transmission Control Protocol) channel in client mode. The “Figure 39” together with:

- the “Difference between Push Mode and Pull Mode”, and
- the “Task Steps”,

explain the remotely administration session flow for SCWS in Push / Pull mode.

Difference between Push Mode and Pull Mode

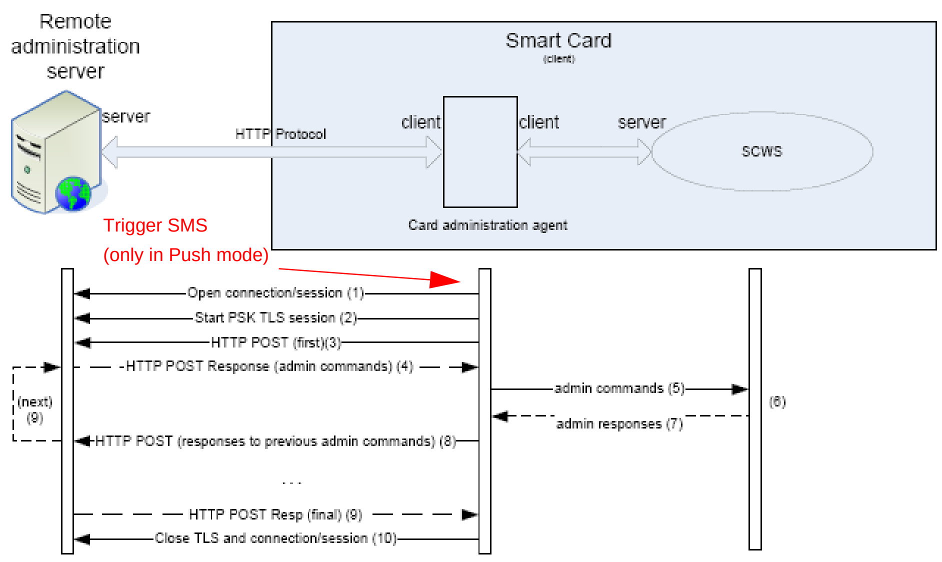
Push Mode

In Push mode, the Server sends a Trigger SMS to the card as shown in the figure below, to trigger the card for an **Open Channel**, in order to establish the communication between the server and the card.

Pull Mode

In Pull Mode, the card initials an **Open Channel**, in order to establish the communication between the server and the card.

Figure 39 - Administration Session Flow


Remote administration server
server
HTTP Protocol
Smart Card
(client)
client
client
server
SCWS
Card administration agent
Trigger SMS
(only in Push mode)
Open connection/session (1)
Start PSK TLS session (2)
HTTP POST (first)(3)
HTTP POST Response (admin commands) (4)
(next)
(9)
admin commands (5)
(6)
admin responses (7)
HTTP POST (responses to previous admin commands) (8)
...
HTTP POST Resp (final) (9)
Close TLS and connection/session (10)

Task Steps

1. Open a BIP channel in TCP client mode with the relevant remote administration server.
2. Use PSK-TLS over this TCP channel to enable mutual authentication, confidentiality and integrity.
3. After the TLS communication channel is established, the card administration agent shall send an HTTP **Post** command in order to get the first administrative command.
4. When receiving the HTTP **Post** from the card administration agent, the remote administration server shall send an HTTP response which encapsulates an HTTP administration command dedicated to the SCWS itself.

5. When receiving the HTTP response for the above HTTP **Post** command, the card administration agent shall forward it to the SCWS.
6. The SCWS shall consider this channel as authenticated by the card administration agent. If the card administration agent is allowed to access the URI involved in the command, it shall process the delivered administration command.
7. After processing the delivered administration command, the SCWS shall deliver the HTTP response back to the card administration agent.
8. The card administration agent shall submit the HTTP response from the SCWS in a new **Post** request to the remote administration server over the TLS secure channel.
9. The remote administration server shall send the next administration command to the card administration agent over the TLS secure channel or send a final response requesting the end of the remote administration session in the **Post** response.
10. If the card administration agent receives a final response from the remote administration server, it shall close the TLS channel and afterwards close the BIP channel.

The OMA SCWS Server Simulator

This section will introduce you the OMA SCWS Server Simulator that integrated in the Simulation Suite V2.6. In this section, you will learn how to:

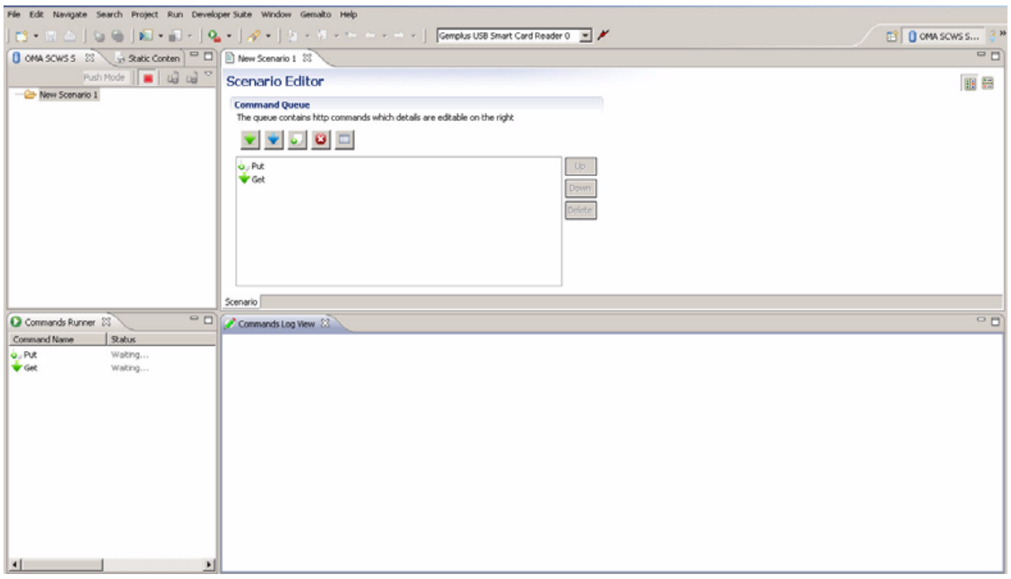
- open the **OMA SCWS Server Simulator** viewer.
- select between **SCWS Server Simulator** and **SMS Trigger**. Also, to configure the port number of https server to be launched and the PSK keys on server side, on the **SCWS Server Simulator** page.
- create a SCWS remote administrative commands scenario.
- start a SCWS server simulation by using either “Push” mode or “Pull” mode.

Menu to Access

From the menu bar, select **Tools** > **OMA SCWS Server Simulator**, or click on **Launch OMA SCWS Server Simulator** button at the toolbar. The **OMA SCWS Server Simulator** viewer is launched.

OMA SCWS Server Simulation Perspective

Figure 40 - The OMA SCWS Server Simulation Perspective



In the OMA SCWS Server Simulation perspective as shown in the figure above, it contains the following panes:

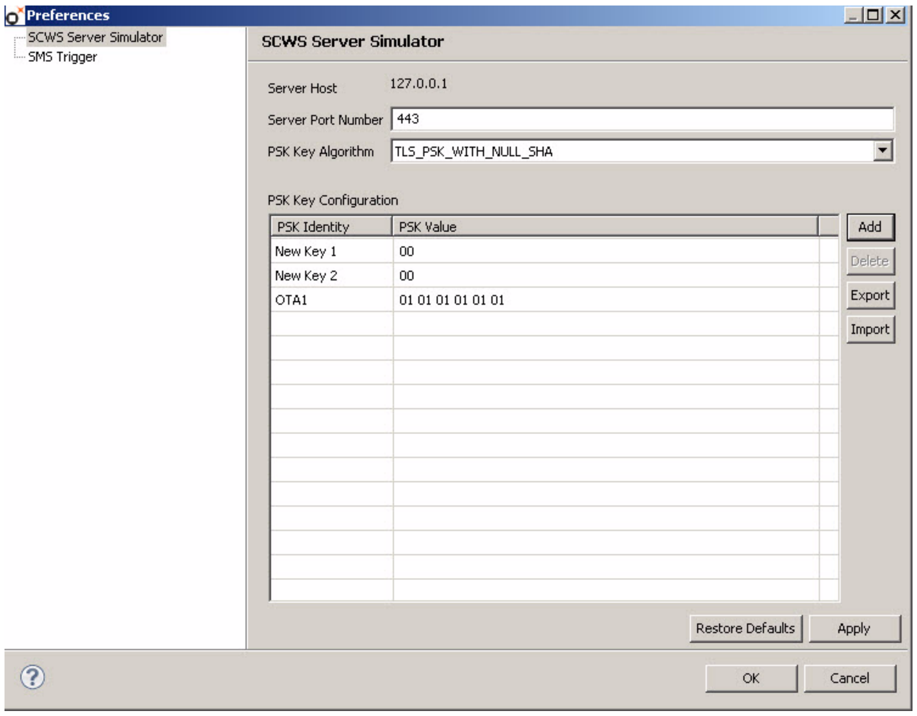
- **OMA SCWS Server Simulator** pane:
  In this window, it provides the features to manage the scenarios in the card.
- **Commands Runner** pane:
  In this window, it provides information of each command processing status which has been executed in the selected scenario.
- **Commands Log View** pane:
  In this window, it provides information of HTTP commands exchange between server and mobile simulator.
- **Scenario Editor** pane:
  In this window, it enables the user to manage the HTTP commands that use in the scenario.

OMA SCWS Server Simulator Connection Configuration - Preference Page

The connection configuration for OMA SCWS Server Simulator can be managed within the **Preference** page.

Click on the **View Menu** and follow by Preferences **Preferences** button will open the **Preferences** window which displays with the **SCWS Server Simulator** panel (default panel) as follows:

Figure 41 - Preferences Page - SCWS Server Simulator (the default panel)



SCWS Server Simulator

Server Host 127.0.0.1

Server Port Number 443

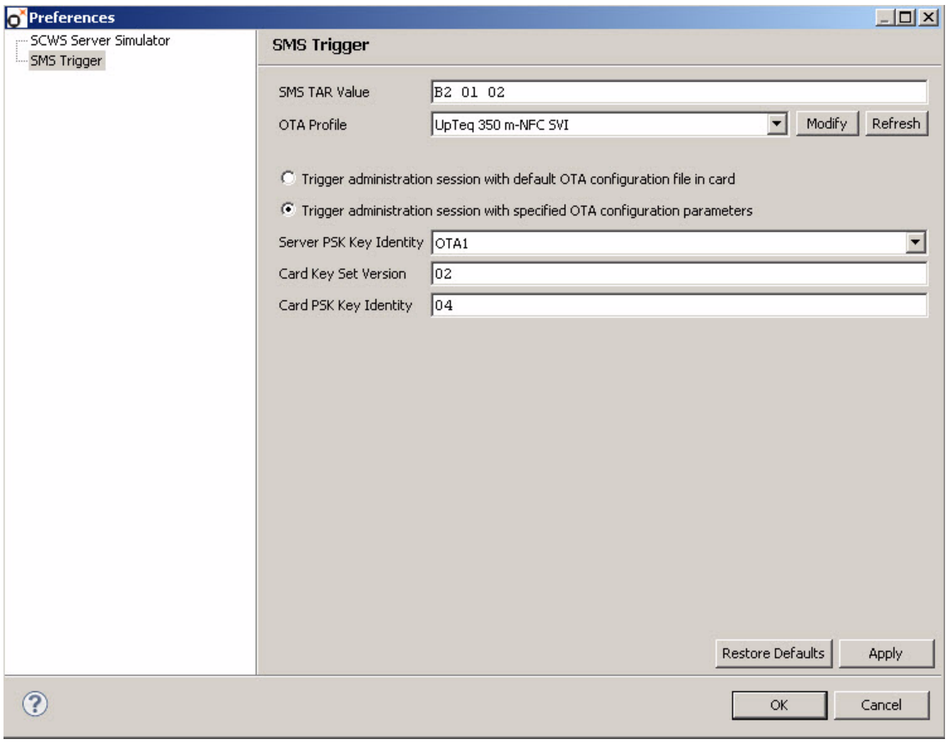
PSK Key Algorithm TLS_PSK_WITH_NULL_SHA

PSK Key Configuration

PSK Identity	PSK Value
New Key 1	00
New Key 2	00
OTA1	01 01 01 01 01 01

User may also select SMS Trigger to turn on the **SMS Trigger** panel as follows:

Figure 42 - Preferences Page - SMS Trigger



Preferences

- SCWS Server Simulator
- SMS Trigger

SMS Trigger

SMS TAR Value	B2 01 02
OTA Profile	UpTeq 350 m-NFC SVI

( ) Trigger administration session with default OTA configuration file in card
[✓] Trigger administration session with specified OTA configuration parameters

Server PSK Key Identity	OTA1
Card Key Set Version	02
Card PSK Key Identity	04

There are three general parts on the **Preferences** window:

- Preference Page Chooser
  User can make a choice to choose between:
  - SCWS Server Simulator—this panel contains connection configuration preference page.
  - SMS Trigger—this panel contains SMS configuration preference page.
- Configuration Panel

This contains the above mentioned two panels—the **SCWS Server Simulator** panel and the **SMS Trigger** panel.

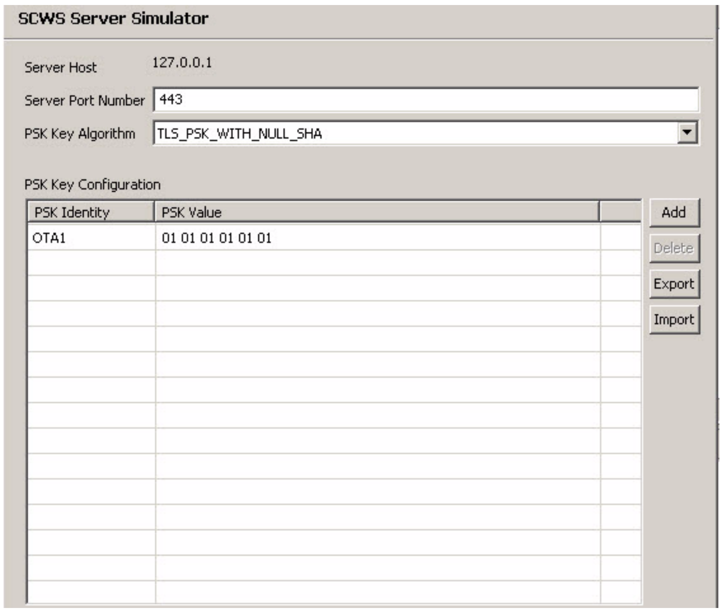
- Standard buttons

The standard buttons contain in the two panels which each of its functionality will be covered as follows:

  - **Restore Defaults**: this button will restore the entire configuration inside a preference page into its default value.
  - **Apply**: this button will apply all the values inside preference page as the new configuration.
  - Help: this ? button will open help content to help user in using the preference page.
  - **OK**: this button will close the preference page and apply all the values inside preference page as the new configuration.
  - **Cancel**: this button will close the preference page without applying any values inside preference page as the new configuration.

Preferences Page - SCWS Server Simulator Panel

Figure 43 - SCWS Server Simulator Panel



SCWS Server Simulator

Server Host 127.0.0.1

Server Port Number 443

PSK Key Algorithm TLS_PSK_WITH_NULL_SHA

PSK Key Configuration

PSK Identity	PSK Value
OTA1	01 01 01 01 01 01

The **SCWS Server Simulator** panel is displayed when **SCWS Server Simulator** is selected. With this panel, user is able to configure the port number of https server to be launched and the PSK keys on server side.

Server Port Number

The default value is 443. Only integer is accepted, otherwise an error message Port value must be integer "Port value must be integer" will be prompted at the top of the panel.

PSK Key Algorithm

Select the algorithm to be used for PSK key from the list.

PSK Key Configuration

In the **PSK Key Configuration** field, user is able to edit the value of PSK Identity and PSK Value by clicking on corresponding column.

In case, the modified PSK Identity with a value is not unique, an error message "PSK Key Identity has to be unique. Fail to insert identity with value: [PSK Identity]" will be displayed on title.

Add Button

At any time, user is able to add a row into **PSK Key Configuration** with the default PSK Identity and value by clicking the **Add** button.

Delete Button

At any time, user is able to delete any row on **PSK Key Configuration** by selecting the row and click **Delete**.

This button is enabled only if user has chosen a row in **PSK Key Configuration** table.

Export Button

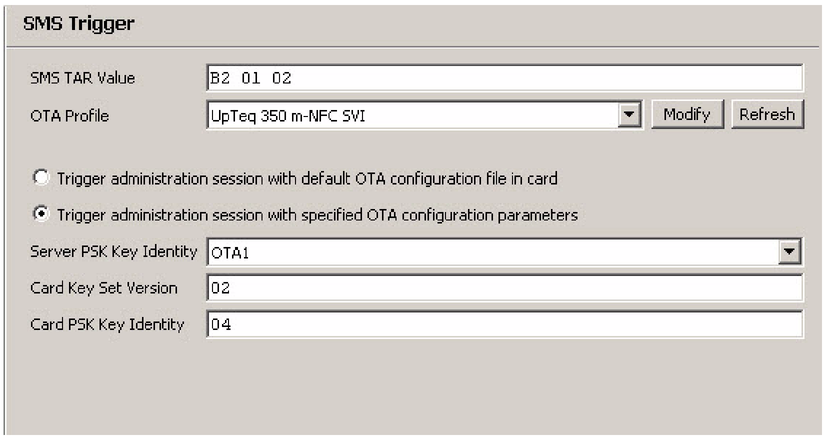
Click the **Export** button and choose a file to save the PSK Key Configuration. The file is saved with keystore extension.

Import Button

Click the **Import** button and choose a file to import. The PSK data from the imported file will be saved in the **PSK Key Configuration**.

Preferences Page - SMS Trigger Panel

Figure 44 - SMS Trigger Panel


SMS Trigger

SMS TAR Value: B2 01 02

OTA Profile: UpTeq 350 m-NFC SVI

( ) Trigger administration session with default OTA configuration file in card

(●) Trigger administration session with specified OTA configuration parameters

Server PSK Key Identity: OTA1

Card Key Set Version: 02

Card PSK Key Identity: 04

The **SMS Trigger** panel is displayed when **SMS Trigger** is selected. With this panel, user is able to specify the details setting on SMS formatting and data content.

SMS TAR Value

The default value is B2 01 02. Only three hexadecimal bytes values are accepted, otherwise an error message “TAR value should be 3 bytes” will be prompted.

OTA Profile

The OTA profile is used to format the SMS (refer to 03.40 and 03.48 standards).

The default value is SCWS Remote Admin Sample. This value can be modified by the **Modify** button.

Modify Button

This button will open the **OTA Profile Manager** dialogue to configure the OTA profile files.

Refresh Button

Click this button to refresh the OTA profile list provided in **OTA Profile** combo box.

Trigger administration session with default configuration resource in card

By default, this is checked, and the tool will send OMA SCWS configuration parameters which contains the settings in the OMA SCWS server simulator to the card. The card will use the parameters directly to perform the connection with remote server simulator.

If this radio button is selected, the tool will send the default OMA SCWS configuration parameters (81 00) to the card through SMS, and the card will perform the remote SCWS admin communication by using the configuration parameters which is defined in the default profile in the card. See “Figure 31 - OTA Configuration Editor Window” under the section “OTA Configuration Interpreter and Editor” on page 39 for more details of the parameters.

Trigger administration session with specified OTA configuration parameters

By default, this is unchecked. If this radio button is selected, the **Server PSK Key Identity**, **Card Key Set Version** and **Card PSK Key Identity** text boxes will be enabled. The tool will send the customized SCWS administration configuration parameters to card through SMS and the card will perform the remote SCWS admin communication by using the configuration parameters in the SMS data.

Server PSK Key Identity

In the **Server PSK Key Identity** field, user is able to select the key identity from the drop down list which is get from the available PSK key database of server simulator.

Card Key Set Version

The default value is 02 and only one hexadecimal byte value is accepted.
This value indicates the key set version (number) which contains the PSK key to be used.

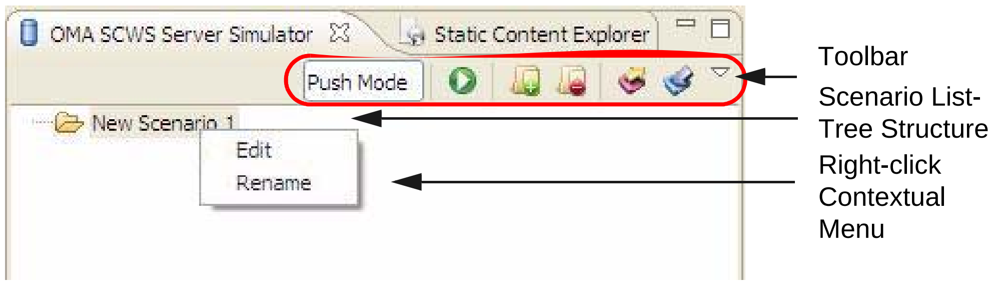
Card PSK Key Identity

The default value is 04 and only one hexadecimal byte value is accepted.
This value indicates which key inside the key set will be used as PSK key on the card.

OMA SCWS Server Simulator Pane

This displays the **OMA SCWS Server Simulator** view in Eclipse workbench.

Figure 45 - OMA SCWS Server Simulator View


OMA SCWS Server Simulator
Static Content Explorer
Push Mode
New Scenario 1
Edit
Rename
Toolbar
Scenario List-Tree Structure
Right-click Contextual Menu

OMA SCWS Server Simulator - Scenario Management

The toolbar for scenario management is located at the top of the **OMA SCWS Server Simulator** pane, and the contextual menu–**Edit** and **Rename**, pop-up when right-click on any of the selected scenario in the Scenario List.

The following contents introduce each button and menu in the **OMA SCWS Server Simulator** pane.

Scenario List in OMA SCWS Server Simulator

This window lists all the scenarios contained in the card. Double click on the corresponding scenario tree item will open **Scenario Editor** window of the selected scenario as described in “Scenario Editor Pane” on page 71.

Push / Pull mode enabler in OMA SCWS Server Simulator

The Push Mode **Push Mode** button is a toggle button to determine the mode—”Push” or “Pull” that the server should work on. When the button is pressed (the pressed-down Push Mode button), the server will work in push mode–that is, the server will send SMS trigger to the mobile simulator each time a scenario is being executed.

Note: If user uses the SMS push mode, the **SMS Trigger** preferences page allows user to specify the details setting on SMS formatting and data content. If user uses the pulling mode (when the Push Mode **Push Mode** button is not pressed), then user will need to go to Static Content Explore to set the SCWS remote admin parameters which is defined in the default profile in the card. See “Figure 31 - OTA Configuration Editor Window” under the section“OTA Configuration Interpreter and Editor” on page 39 for more details of the parameters.

Execute the scenario in OMA SCWS Server Simulator

The **Execute** button to execute the scenario. This will start a HTTPs server and launch mobile simulator (with BIP function) automatically. The SMS trigger will be sent if it is in Push mode.

Add new scenario to OMA SCWS Server Simulator

The **Add Scenario** button will create an new scenario and add it in the Scenario List. The new scenario will be named New Scenario X where X is the available number from current Scenario List.

Delete scenario from OMA SCWS Server Simulator

The **Delete Scenario** button will delete the selected scenario in the Scenario List. Once the scenario is deleted, the button will be disable. It will be enabled again when a new scenario is being selected in the Scenario List.

Export scenario from OMA SCWS Server Simulator

The **Export Scenario** button will export the whole database of scenario in the Scenario List.

Import scenario from OMA SCWS Server Simulator

The **Import Scenario** button will import the whole database of scenario into the Scenario List. Once the importing is done, the previous database of scenario in the Scenario List will be totally replaced by the new imported database of scenario.

Preferences page window in OMA SCWS Server Simulator

The **View Menu** button will open preferences page window as described in “OMA SCWS Server Simulator Connection Configuration - Preference Page” on page 64.

Edit the scenario in OMA SCWS Server Simulator

The **Edit** button will open **Scenario Editor** window of the selected scenario as described in “Scenario Editor Pane” on page 71.

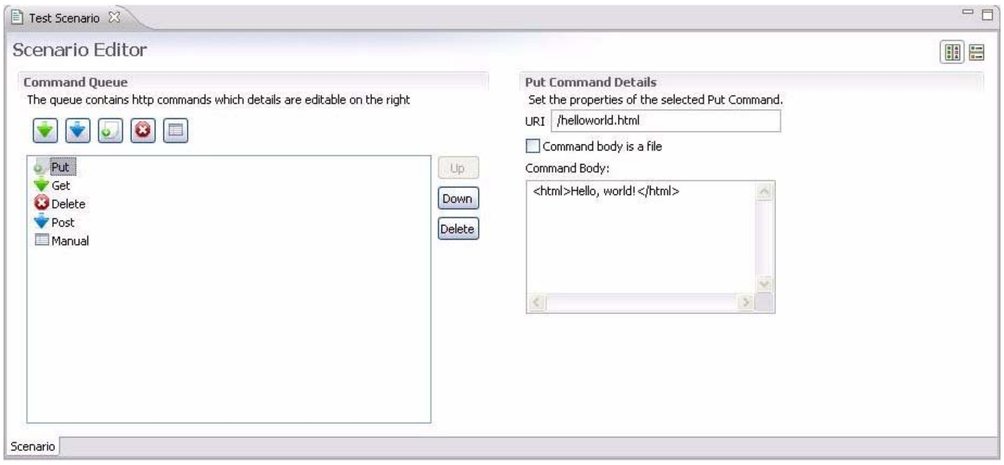
Rename the scenario in OMA SCWS Server Simulator

The **Rename** button will enable user to rename the selected scenario in the Scenario List. By renaming the scenario, the corresponding scenario editor's title will also be updated.

Scenario Editor Pane

Double click on any scenario tree item in Scenario List will display the corresponding **Scenario Editor** view in Eclipse workbench.

Figure 46 - Scenario Editor View


Test Scenario

Scenario Editor

Command Queue

The queue contains http commands which details are editable on the right

Put
Get
Delete
Post
Manual

Up
Down
Delete

Put Command Details

Set the properties of the selected Put Command.

URI /helloworld.html

Command body is a file

Command Body:

```
<html>Hello, world!</html>
```

Scenario

Scenario Editor- Command Editor

The **Scenario Editor** consists of two views:

- Command Queue view:
  This view contains HTTP commands which used to manage the scenario.
- Command Details view:
  This view displays the details of the selected command, and enable you to edit the parameters of the command.
  Clicking on the command in the list will display its corresponding command details in this view.

These two views can be displayed side by side vertically or horizontally. Click on the button to have a vertical orientation, and button to have a horizontal oriental.

The following contents introduce the command administrative buttons (Up, Down, Delete), each HTTP command and its corresponding command details in the **Scenario Editor** pane.

The Command Administrative Buttons

Up button

The Up button is used to move the selected command one position above its current position in the list.
This button is enabled only if the selected command is not at the top position in the list.

Down button

The Down button is used to move the selected command one position below its current position in the list.
This button is enabled only if the selected command is not at the bottom position in the list.

Delete button

The Delete button is used to remove the selected command from the list. This button is enabled only if a command is selected in the list.

The HTTP Commands and the Command Details

Get command

Click this button will add **Get** command into Commands List and display Get Command Editor on Command Details. In the Command Details view, user is able to input the URI value.

Post command

Click this button will add **Post** command into Commands List and display Post Command Editor on Command Details. In the Command Details view, user is able to input the URI value and the Command Body value.

Put command

Click this button will add **Put** command into Commands List and display Put Command Editor on Command Details. In the Command Details view, user is able to input the URI value. The **Command body is a file** is checked by default, click **Browse** to locate the file, otherwise, input the Command Body value manually, if the box is not checked.

Delete command

Click this button will add **Delete** command into Commands List and display Delete Command Editor on Command Details. In the Command Details view, user is able to input the URI value.

Menu command

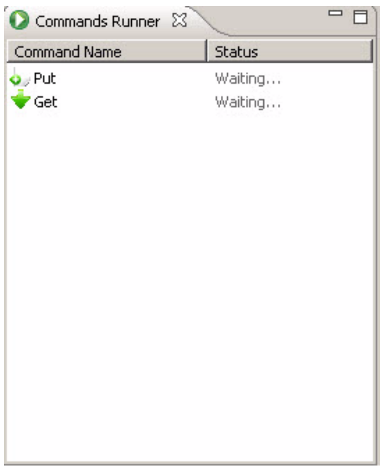
Click this button will add **Manual** command into Commands List and display Manual Command Editor on Command Details. In the Command Details view, user is able to manually input his custom Http command.

Note: Click **File** > **Save** or **Ctrl** + **S** keys from your keyboard, to save the updates that have done within the **Scenario Editor** pane before the execution process of the scenario. This is to ensure the most updated command status is reflected for the executed scenario.

Commands Runner Pane

This displays the **Commands Runner** view in Eclipse workbench which enable user to check the processing status of each command for the scenario that has been selected for execution.

Figure 47 - Command Runner View



Command Name	Status
Put	Waiting...
Get	Waiting...

Command Runner - Command Executor

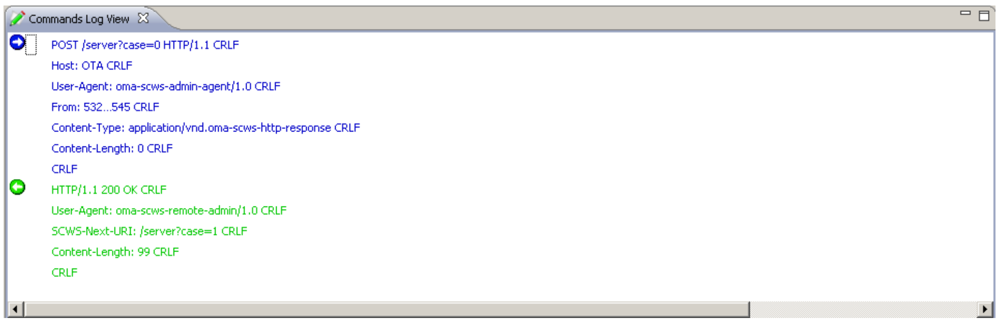
The command status can be accessed by executing any scenario through clicking the **Execute** button on the **OMA SCWS Server Simulator** pane. The list of command status will be updated and displayed on the **Command Runner** view.

Note: Please click **File** > **Save** or **Ctrl** + **S** keys from your keyboard, to save the updates that have done within the **Scenario Editor** pane before the execution process of the scenario. This is to ensure the most updated command status is reflected for the executed scenario.

Commands Log View Pane

This displays the **Commands Log View** in workbench which provides information of HTTP commands exchange between server and mobile simulator.

Figure 48 - Command Log View



```
POST /server?case=0 HTTP/1.1 CRLF
Host: OTA CRLF
User-Agent: oma-scws-admin-agent/1.0 CRLF
From: 532...545 CRLF
Content-Type: application/vnd.oma-scws-http-response CRLF
Content-Length: 0 CRLF
CRLF
HTTP/1.1 200 OK CRLF
User-Agent: oma-scws-remote-admin/1.0 CRLF
SCWS-Next-URI: /server?case=1 CRLF
Content-Length: 99 CRLF
CRLF
```

Command Log View - Command Log Viewer

The command exchange status can be accessed by executing any scenario through clicking the **Execute** button on the **OMA SCWS Server Simulator** pane. The list of commands exchange between server and mobile simulator will be displayed on the **Command Log View**.

Note: Please click **File** > **Save** or **Ctrl** + **S** keys from your keyboard, to save the updates that have done within the **Scenario Editor** pane before the execution process of the scenario. This is to ensure the most updated command status is reflected for the executed scenario.

The File System Editor

What is the File System Editor?

This tool enables you to view the file structure in a card or a card simulator and the properties and access conditions for any file. In addition it can be used to perform certain commands on files.

You can launch the File System editor either from within the JCardManager, or from the **Developer Suite** menu in Eclipse.

To start the File System Editor from JCardManager:

1 Select the card simulator or reader in the **Terminal** list in the toolbar.

2 Select the card type in the tree structure in the left pane.

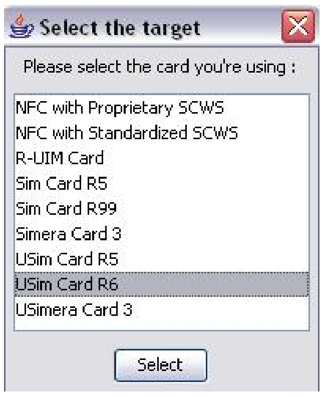
3 Click the **File System Editor** button in the toolbar.

To start the File System Editor from Eclipse:

1 Choose **File System Editor** from the **Developer Suite** menu.

2 In the Select the Target window that appears, choose the type of card you are using and click **Select**:

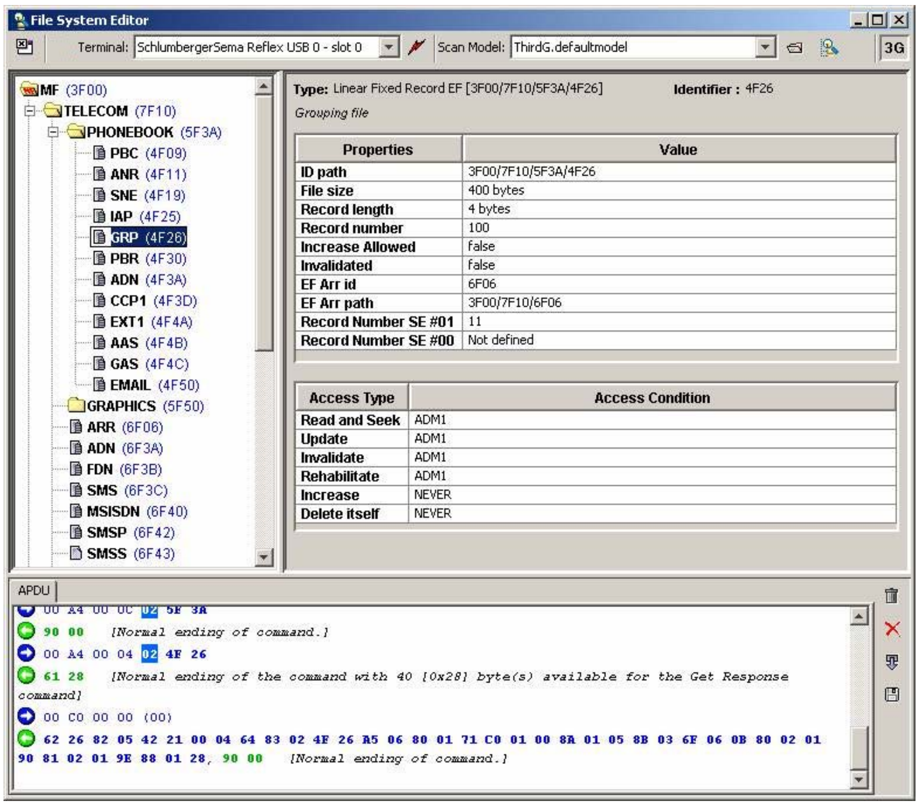
Figure 49 - The Select the Target Window



The File System Editor Interface

The File System Editor window is shown in the following figure:

Figure 50 - File System Editor



File System Editor
Terminal: SchlumbergerSema Reflex USB 0 - slot 0
Scan Model: ThirdG.defaultmodel
3G
APDU

MF (3F00)
- TELECOM (7F10)
  - PHONEBOOK (5F3A)
    - PBC (4F09)
    - ANR (4F11)
    - SNE (4F19)
    - IAP (4F25)
    - GRP (4F26)
    - PBR (4F30)
    - ADN (4F3A)
    - CCP1 (4F3D)
    - EXT1 (4F4A)
    - AAS (4F4B)
    - GAS (4F4C)
    - EMAIL (4F50)
  - GRAPHICS (5F50)
  - ARR (6F06)
  - ADN (6F3A)
  - FDN (6F3B)
  - SMS (6F3C)
  - MSISDN (6F40)
  - SMSP (6F42)
  - SMSS (6F43)

Type: Linear Fixed Record EF [3F00/7F10/5F3A/4F26] **Identifier :** 4F26

Grouping file

Properties	Value
ID path	3F00/7F10/5F3A/4F26
File size	400 bytes
Record length	4 bytes
Record number	100
Increase Allowed	false
Invalidated	false
EF Arr id	6F06
EF Arr path	3F00/7F10/6F06
Record Number SE #01	11
Record Number SE #00	Not defined

Access Type	Access Condition
Read and Seek	ADM1
Update	ADM1
Invalidate	ADM1
Rehabilitate	ADM1
Increase	NEVER
Delete itself	NEVER

```
00 A4 00 0C 02 5F 3A
90 00   [Normal ending of command.]
00 A4 00 04 02 4F 26
61 28   [Normal ending of the command with 40 [0x28] byte(s) available for the Get Response
command]
00 C0 00 00 (00)
62 26 82 05 42 21 00 04 64 83 02 4F 26 A5 06 80 01 71 C0 01 00 8A 01 05 8B 03 6F 06 0B 80 02 01
90 81 02 01 9E 88 01 28, 90 00   [Normal ending of command.]
```

Depending on the **Scan Model** you select, the File System Editor operates in 2G or 3G mode. The current mode is displayed to the right of the toolbar.

Select a file in left pane to display the following properties in the right pane:

- Type, identifier and very brief description
- Properties
- Access type
- PINs and their statuses, that is, whether or not they have been successfully presented. This last feature applies to the MF and DFs only.

Contextual Commands

You can perform certain commands on a file in the file system editor. The commands available depend on the type of file. The following table describes the commands available for each file type:

Table 6 - Contextual Commands

Command	Description	File type
Scan	Updates the information displayed in the right pane	MF, DF, ADF (3G only)
Select	Makes the selected file the current file	MF, DF, EF, ADF (3G only)
File Access Management: ■ **Verify PIN** ■ **Change PIN** ■ **Enable PIN** ■ **Disable PIN** ■ **Unblock PIN**	Performs the PIN command on all the files in the card.	MF, DF, EF, ADF (3G only)
Create	Creates an EF or DF under the chosen file. **Note:** This command is only available for R6-compliant card profiles (USIM R6 and R-UIM). **Note:** This command is not available for the default scan model: you must create a copy of a default scan model before you are able to create files.	MF, DF, ADF (3G only)
Delete	Deletes the chosen file. **Note:** This command is only available for R6-compliant card profiles (USIM R6 and R-UIM. **Note:** This command is not available for the default scan model: you must create a copy of a default scan model before you are able to delete files.	DF, EF, ADF (3G only)
Update Record/Binary	Updates a record or data in a data file	EF

To perform a contextual command:

1 Choose the file in the left pane on which you want to perform the command.

2 Right–click and select the command from the contextual menu.

3 For **Create**, **PIN** commands and **Update** commands, complete the dialog box for the command and click the button with the command name, for example **Update Binary**.

4 Click **Close**.

The pane at the bottom of the File System Editor displays the APDU code for the command.

For details about individual commands, refer to the document *ETSI TS 102 221 V7.0.0 (2004-12); Technical Specification; Smart cards; UICC-Terminal interface; Physical and logical characteristics (Release 7)*, available at www.etsi.org.

The Smart Card Reader Manager

The Gemalto Smart Card Reader Manager is used for protocol switching among the supported protocols.

To launch a Smart Card Reader Manager:

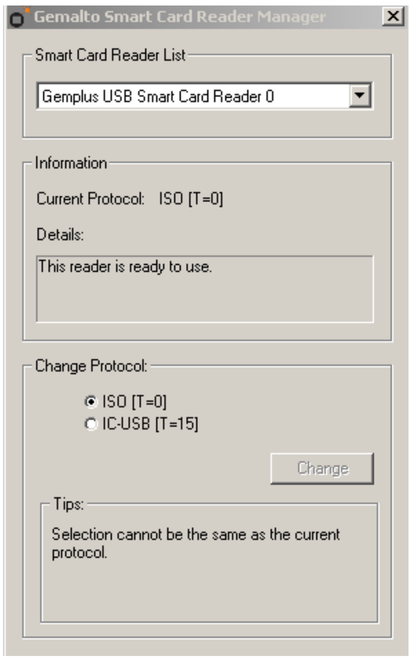
The Smart Card Reader Manager can be launch from **Start** > **All Programs** > **Gemalto** > **Smart Card Reader Manager > SCard Reader Manager**.

The **Gemalto Smart Card Reader Manager** icon appears on the status bar shows the successful launching. Double click on the icon to start using the Gemalto Smart Card Reader Manager.

To use Smart Card Reader Manager:

Double click icon, the Smart Card Reader Manager window as in "Figure 51" appears.

Figure 51 - Smart Card Reader Manager Window-Before Switching Protocol.


Gemalto Smart Card Reader Manager

Smart Card Reader List

Gemplus USB Smart Card Reader 0

Information

Current Protocol: ISO [T=0]

Details:

This reader is ready to use.

Change Protocol:

[✓ ISO [T=0]]
IC-USB [T=15]

Tips:

Selection cannot be the same as the current protocol.

This window shows the state of the reader, such as:

- the status of the card in the reader,
- the current protocol in use,
- the list of supported protocols.

Select a reader from the drop-down box, the information of reader or the status of the card in the corresponding reader is displayed in the **Details** field of the **Information** area.

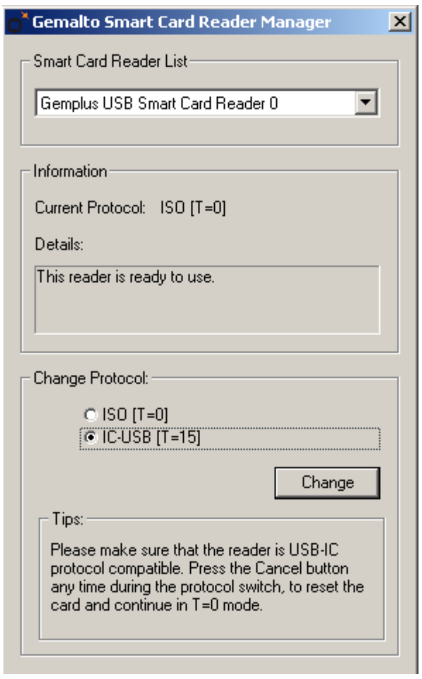
Note: The **Change Protocol** section of the window will not be displayed if there is no card in the reader.

Switching Protocol from the Supported Protocols

Smart Card Reader Manager enables protocol switching from the list of supported protocols.

Select the protocol you wanted to switch to in the **Change Protocol** area, press the **Change** button to change to the new protocol (for example, switching from ISO [T=0] to IC-USB [T=15]).

Figure 52 - Smart Card Reader Manager Window-Select on New Protocol.



Gemalto Smart Card Reader Manager

Smart Card Reader List

Gemplus USB Smart Card Reader 0

Information

Current Protocol: ISO [T=0]

Details:

This reader is ready to use.

Change Protocol:

ISO [T=0]
[✓ IC-USB [T=15]]

Change

Tips:

Please make sure that the reader is USB-IC protocol compatible. Press the Cancel button any time during the protocol switch, to reset the card and continue in T=0 mode.

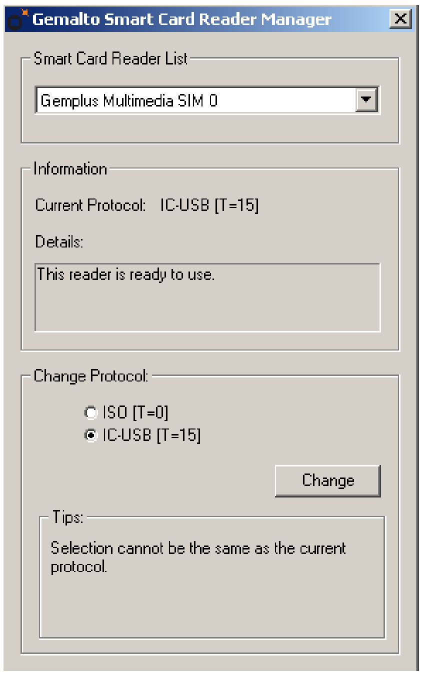
Note: The **Change** button will be disabled if the selected protocol is the same as the current one.

Once the **Change** button is pressed, a progress dialog will appear until the protocol switch is successful and the new reader is available in the reader list.

Pressing the **Cancel** button will abort the PPS negotiation, this will result in the card being reset and switching back to T=0 protocol.

Switching back from IC-USB [T=15] to ISO [T=0] is to be done by re-inserting the card, and therefore the **Change** button will be disabled.

Figure 53 - Smart Card Reader Manager Window-After Protocol Switching.


Gemalto Smart Card Reader Manager

Smart Card Reader List

Gemplus Multimedia SIM 0

Information

Current Protocol: IC-USB [T=15]

Details:

This reader is ready to use.

Change Protocol:

ISO [T=0] [✓ IC-USB [T=15]]

Change

Tips:

Selection cannot be the same as the current protocol.

If the protocol switch does not succeed, a message will pop up to indicate this.

Protocol switch may not succeed if the selected protocol is not supported by the card or the reader that you are using. For example: the IC-USB protocol is not supported by the GemPC Twin readers

Universal Tracer

The Universal Tracer can be used to trace the APDU exchanges between different devices or simulators.

Working with the Universal Tracer

The Universal Tracer enables you to trace APDU communication exchanges between a PC application and a PC/SC-compatible smartcard reader or card simulator to debug or check compliance with established standards. When used with a special USB mobile-SIM connector (a MSC device), the Universal Tracer can also track the APDU exchanges between a mobile equipment (ME) and a (U)SIM card. These communication exchanges are subsequently logged and interpreted.

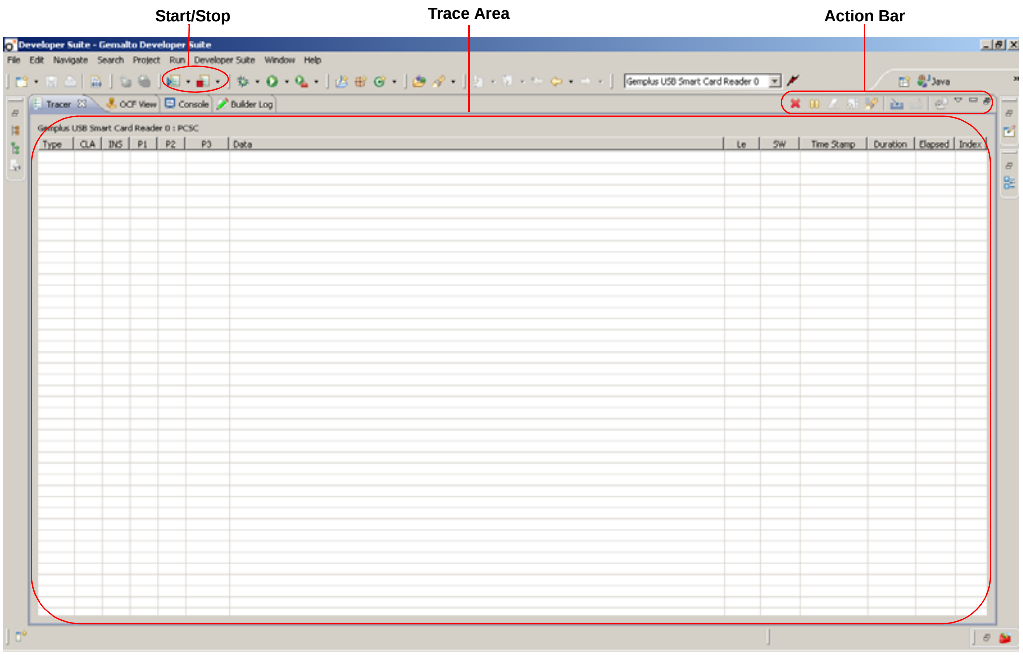
The Universal Tracer user interface consists of three main parts:

- Trace Area
- Action Bar
- Start/Stop

The **Trace Area** displays the traced data and the **Action Bar** contains the buttons that allows you to perform different operations on the trace. Use the **Start** button on the toolbar to initiate a trace session and **Stop** to end the trace session.

The layout of the Universal Tracer is as follows:

Figure 54 - The Universal Tracer View


Start/Stop
Trace Area
Action Bar
Developer Suite - Gemalto Developer Suite
File Edit Navigate Search Project Run Developer Suite Window Help
Gemplus USB Smart Card Reader 0
Java
Tracer
OCF View
Console
Builder Log
Gemplus USB Smart Card Reader 0 : PCSC
Type CLA INS P1 P2 P3 Data Le SW Time Stamp Duration Elapsed Index

Using the Tracer

Before you can begin working with the Universal Tracer, you must define the data source you want to work with. The media can be connected to your PC via the following sources:

- A mobile-SIM connector (MSC)
- A PC/SC-compatible smartcard reader
- A card simulator

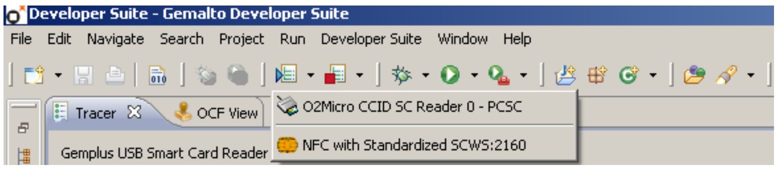
Note: To work with MSC source, additional hardware is needed. Please contact Hotline for more details.

In the Universal Tracer, the format of the respective data source is displayed in the following order:

1. MSC device	USB Mobile SIM Connector - COM <Port #>
	For example, USB Mobile SIM Connector - COM5
2. PC/SC card reader	<Application Name> - <Card Reader Name> <Port #>
	For example, O2 Micro - CCID SC Reader - 0
3. Card simulator	<Card Simulator Name>:<Port #>
	For example, NFC with Standardized SCWS:2160

To see the data sources that are connected to your PC, click the drop-down list on .

Figure 55 - An Example of Connected Data Sources


Developer Suite - Gemalto Developer Suite
File Edit Navigate Search Project Run Developer Suite Window Help
Tracer
OCF View
Gemplus USB Smart Card Reader
O2Micro CCID SC Reader 0 - PCSC
NFC with Standardized SCWS:2160

To start a trace session:

1 Connect the data source.

2 Run a PC application to be used for tracking the APDU exchanges between the media and the PC application.

3 Click to start the trace session. Perform the sequence of commands to be traced in the PC application. The trace information and any returned APDU response are displayed in the Trace area.

4 To interpret the trace information, click and the interpreted trace file shows the APDU commands that are sent to the card and the responses returned by the card.

5 At any time, you can:

- Click on the toolbar to temporarily suspend the trace session. The trace session is suspended until is pressed again to restart tracing.
- Either click on the toolbar to stop tracing and close the trace session.

6 Enter the name of the new trace and click to save the trace information.

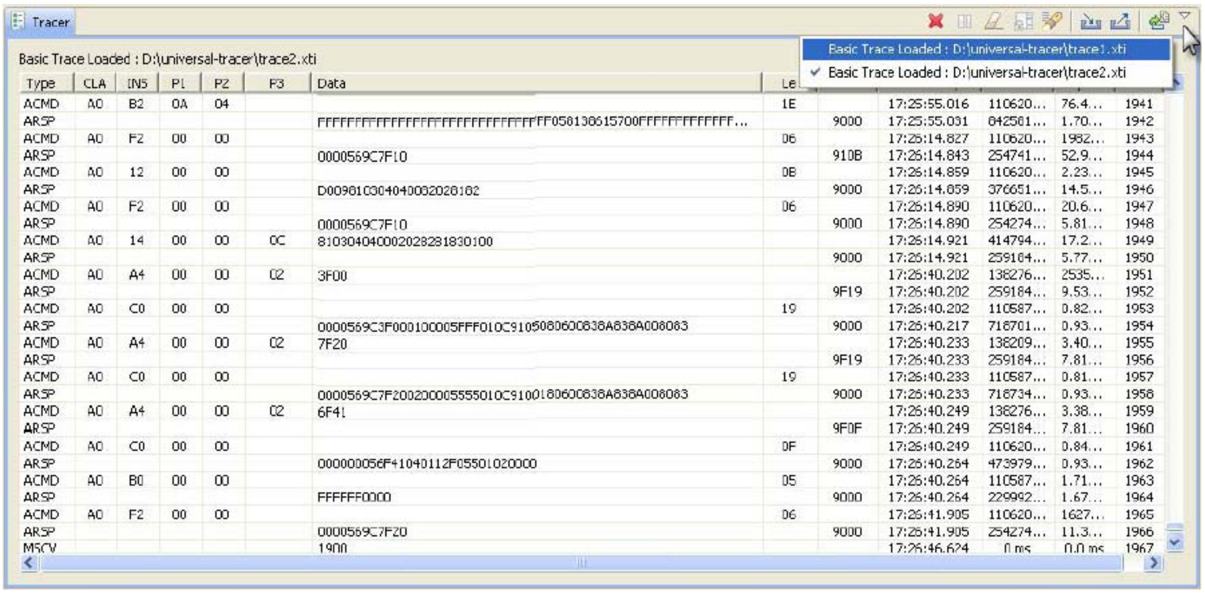
7 To close the trace session at any time, click .

Alternatively, you can load a trace file (in .xti or .per format) to track a pre-recorded session by clicking .

There are options to copy, extract selected items, clear a trace and search a trace via right-clicking any row in the trace session or in the interpreted mode. For more information about the use of these options, click the **Help** button in the respective contextual help window.

To switch between the different tracer windows, click the drop-down list on the right-hand side of the Tracer window and click the preferred window as shown in the following.

Figure 56 - Switching Between Tracer Windows


Tracer
Basic Trace Loaded : D:\universal-tracer\trace1.xti
✓ Basic Trace Loaded : D:\universal-tracer\trace2.xti

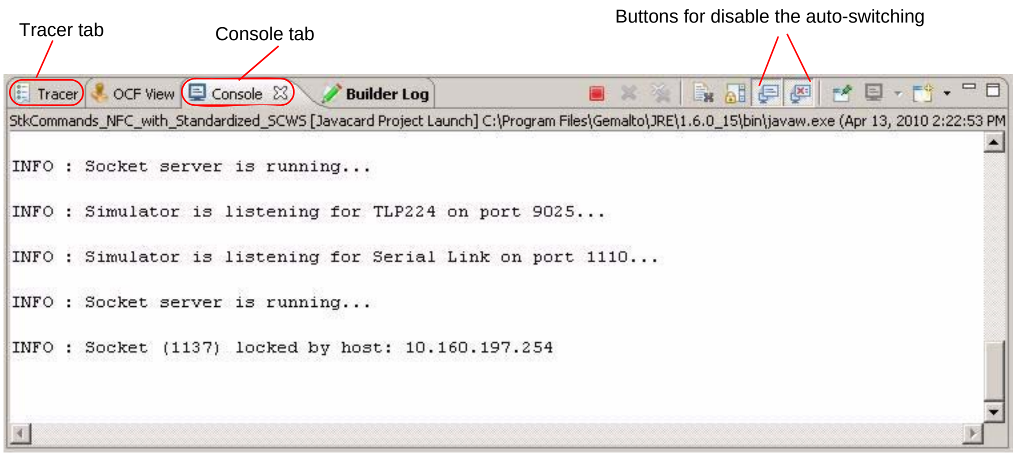
Basic Trace Loaded : D:\universal-tracer\trace2.xti

Type	CLA	INS	P1	P2	P3	Data	Le					
ACMD	A0	B2	0A	04			1E		17:25:55.016	110620...	76.4...	1941
ARSP						FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF058138615700FFFFFFFFFFFF...		9000	17:25:55.031	842581...	1.70...	1942
ACMD	A0	F2	00	00			06		17:26:14.827	110620...	1982...	1943
ARSP						0000569C7F10		910B	17:26:14.843	254741...	52.9...	1944
ACMD	A0	12	00	00			0B		17:26:14.859	110620...	2.23...	1945
ARSP						D009810304040082028182		9000	17:26:14.859	376651...	14.5...	1946
ACMD	A0	F2	00	00			06		17:26:14.890	110620...	20.6...	1947
ARSP						0000569C7F10		9000	17:26:14.890	254274...	5.81...	1948
ACMD	A0	14	00	00	0C	810304040002028281830100			17:26:14.921	414794...	17.2...	1949
ARSP								9000	17:26:14.921	259184...	5.77...	1950
ACMD	A0	A4	00	00	02	3F00			17:26:40.202	138276...	2535...	1951
ARSP								9F19	17:26:40.202	259184...	9.53...	1952
ACMD	A0	C0	00	00			19		17:26:40.202	110587...	0.82...	1953
ARSP						0000569C3F000100005FFF010C9105080600838A838A008083		9000	17:26:40.217	718701...	0.93...	1954
ACMD	A0	A4	00	00	02	7F20			17:26:40.233	138209...	3.40...	1955
ARSP								9F19	17:26:40.233	259184...	7.81...	1956
ACMD	A0	C0	00	00			19		17:26:40.233	110587...	0.81...	1957
ARSP						0000569C7F200200005555010C9100180600838A838A008083		9000	17:26:40.233	718734...	0.93...	1958
ACMD	A0	A4	00	00	02	6F41			17:26:40.249	138276...	3.38...	1959
ARSP								9F0F	17:26:40.249	259184...	7.81...	1960
ACMD	A0	C0	00	00			0F		17:26:40.249	110620...	0.84...	1961
ARSP						000000056F41040112F05501020000		9000	17:26:40.264	473979...	0.93...	1962
ACMD	A0	B0	00	00			05		17:26:40.264	110587...	1.71...	1963
ARSP						FFFFFF0000		9000	17:26:40.264	229992...	1.67...	1964
ACMD	A0	F2	00	00			06		17:26:41.905	110620...	1627...	1965
ARSP						0000569C7F20		9000	17:26:41.905	254274...	11.3...	1966
MSCV						1900			17:26:46.624	0 ms	0.0 ms	1967

Disable Auto-Switching to Console View from Tracer View

User may experience that the Universal Tracer tracing view (**Tracer tab** is enabled) suddenly switched to console view (**Console tab** is enabled) and need to switch back to tracing view tab. This is a default behavior in any Eclipse based application, due to the console view receives some output to be displayed during the process. To prevent this happens, user can disable the auto-switching by clicking the toggle buttons and on the console view. Button is the toggle to show console when there is standard out changes and button is the toggle to show console when standard error changes. By disabling these two toggle buttons, the console view will not be automatically switched from tracing view.

Figure 57 - Example of Disable Auto-Switching to Console View



```
StkCommands_NFC_with_Standardized_SCWS [Javacard Project Launch] C:\Program Files\Gemalto\JRE\1.6.0_15\bin\javaw.exe (Apr 13, 2010 2:22:53 PM

INFO : Socket server is running...

INFO : Simulator is listening for TLP224 on port 9025...

INFO : Simulator is listening for Serial Link on port 1110...

INFO : Socket server is running...

INFO : Socket (1137) locked by host: 10.160.197.254
```

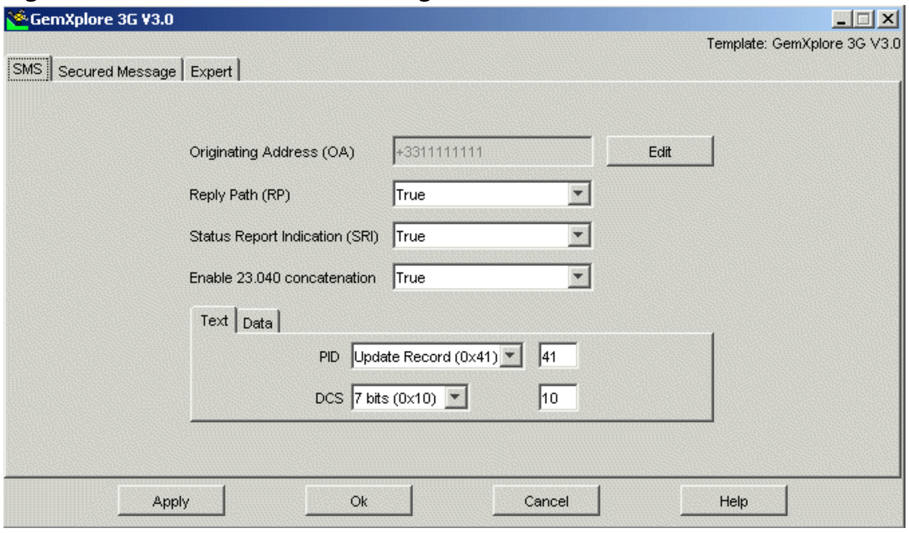
Managing OTA Profiles

This appendix describes how to configure OTA profiles in the OTA Profile Manager. To start the OTA Profile Manager, select a card profile in the **Target** area of the Application Manager and click .

The OTA Profile Manager

The main window of the OTA Profile Manager is shown in “Figure 58”:

Figure 58 - The OTA Profile Manager Window



GemXplore 3G V3.0

Template: GemXplore 3G V3.0

Originating Address (OA): +3311111111

Reply Path (RP): True

Status Report Indication (SRI): True

Enable 23.040 concatenation: True

PID: Update Record (0x41) 41

DCS: 7 bits (0x10) 10

This window allows you to modify the selected OTA profile, in this example the “GemXplore 3G V3.0” displayed on the window’s title bar.

The card template upon which the profile is based is displayed in the top right-hand corner of the window.

An OTA profile’s parameters are spread across three property sheets:

- **SMS**. These parameters select optional features of SMS messaging, such as whether to implement concatenation, use a reply path, or request a status report.

  These options are described in detail in “SMS Options” on page 88.

- **Secured Message**. These parameters determine which optional values of the 3GPP TS 23.048 specification to use when formatting command and response packets.

  These options are described in detail in “Secured Message Options” on page 90.

- **Expert**. These parameters are intended for use by expert users only, and concern the format of an SMS message's concatenation header, command packet and response packet headers.

  These parameters are described in "Expert Options" on page 95.

Optionally, click **Apply** to save the current values of parameters on the selected property sheet.

Click **OK** to save the current values of all parameters on all property sheets and close the OTA parameters window.

SMS Options

The **SMS** property sheet contains options relating to the Short Message Service (SMS). The default values displayed are read from the template upon which this card profile is based.

You can specify values for the following parameters:

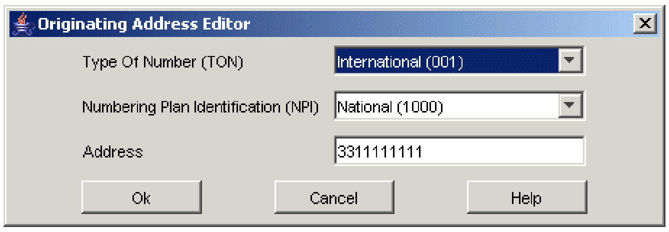
Originating Address

It is usual for the receiving entity to implement a number of security mechanisms, which may include a check that the message was sent by an authorized application provider. This is done by checking the "originating address" (TP-OA field) included in the message's OTA header.

Note: The originating address is the network address (MSISDN) of the message's originator, that is, the sending entity, not that of the SMSC that forwards the message to the SIM card.

To modify the default originating address, click **Edit** to display the **Originating Address Editor** window:

Figure 59 - The Originating Address Editor Window


Originating Address Editor

Type Of Number (TON): International (001)

Numbering Plan Identification (NPI): National (1000)

Address: 3311111111

Specify the **Type of Number (TON)**, **Numbering Plan Identification (NPI)**, and **Address** components of the originating address:

- **TON**. Indicates the format of a number. The most commonly used settings are:
  - **National number**. A sequence of digits, usually beginning with a "0".
  - **International number**. In this format, the number starts with a "+" followed by the country code, even for national calls. This format is recommended for roaming and international calls, and highly recommended for stored numbers and call forwarding.

Note: The digits in parentheses following the options refer to the normative values of the TON/NPI byte, bits 4 to 6. For example, "001" indicates use of the "International" TON.

- **NPI**. Used to ensure compatibility between numbering plans (for example, for a private network) in which the digit sequences may have a different meaning and structure than in the network’s default numbering plan. The most commonly used setting is “ISDN/Telephone Number Plan”.

- **Address**. A string specifying the number of an entity (service center, mobile station) on the network.

Reply Path (RP)

Specifies whether the receiving entity is requested to reply to the SMS message:

- **True**. A reply is to be sent. The value in the **Originating Address** field is used to route the reply.
- **False**. No reply is to be sent.

Status Report Indication (SRI)

Indicates whether the receiving entity is requested to send a status report:

- **True**. A status report is to be generated and sent.
- **False**. No status report is to be sent.

The SRI is coded as the TP-SRI field of the SMS DELIVER TPDU defined by the 3GPP 23.040 standard.

Enable 23.040 Concatenation

Indicates whether concatenation is to be used.

Concatenation, described in the 3GPP 23.040 specification, provides a mechanism for transmitting command packets containing large blocks of data by breaking the data up into segments and transmitting each segment as a separate SMS message. The segments are then reassembled by the receiving entity (provided that it supports concatenation: not all SIM cards in the field today support concatenation).

Without concatenation, a single SMS message can only contain up to 160 characters of text (where each character is 7 bits according to the 7-bit default alphabet), or up to 140 characters of 8-bit binary data.

Specify either:

- **True**. This message implements concatenation; the receiving entity should expect other messages in the sequence.
- **False**. This message does not implement concatenation.

If you choose to implement concatenation, refer to the **Concatenation** property sheet to set optional values for concatenation; see “Expert Options” on page 95.

Text and Data Format

Use the **Text** and **Data** property sheets to indicate to the receiving entity the method to use to load the text and data contained in the message onto the SIM card:

PID

Select the Protocol Identifier (PID) to use:

- **Update Record**. Download the text or data using an **Update Record** SIM Toolkit command to update the EF$_{SMS}$ elementary file in the SIM card’s GSM file system.

- **Envelope**. Download the text or data by sending an **SMS Point-To-Point Data Download Envelope** command to the SIM card.

Note: Regardless of the method chosen, the capabilities of the mobile equipment determine which method is actually used. For example, the **SMS Point-To-Point Data Download Envelope** command is only supported by Phase 2+ mobile equipment. Phase 2 mobile equipment systematically loads OTA messages using the **Update Record** command, even if the **Envelope** option is selected.

DCS

Select the Data Coding Scheme (DCS) that the receiving entity is to use to encode the text or data. The choice affects the maximum number of characters or bytes that a single SMS message can contain:

- **8-bit.** Unpacked format (eight-bit data), bit 7 is set to 0. Unused bytes are set to FFh. A **Text** message string may contain up to 210 characters. A **Data** message may contain up to 210 bytes.
- **7-bit.** Packed format, the default seven-bit alphabet is used, packed into bytes. A **Text** message string may contain up to 240 characters. A **Data** message may contain up to 210 bytes.
- **UCS-2**. In UCS-2 format, the UCS-2 alphabet is used. In a **Text** message, each character uses two bytes and the string may contain up to 105 characters. A **Data** message may contain 210 bytes.
- **Custom**. The text or data in the message is in a custom format understood by the receiving entity.

Secured Message Options

These parameters determine the 3GPP 23.048 formatting of the message’s command packet. The parameters are further divided into three sub-property sheets: click on the **SPI**, **KIc** or **KID** tabs on the right of the window to access the other property pages.

SPI

The Secured Packet Information (SPI) is a 2-byte field of the message’s command packet indicating:

- In byte 1, the security mechanisms to implement in order to secure the message.
- In byte 2, the settings to build the response packet returned by the receiving entity.

Ciphering

Whether the contents of the message are ciphered:

- **True**. The contents are ciphered
- **False**. The contents are in “clear”.

If ciphering is to be implemented, use the options on the KIc property sheet to configure the ciphering mechanism. Refer to “KIc” on page 92.

Security

Indicates which security mechanism is to be implemented:

- **No RC, CC, or DS**. No security mechanism is used: the text or data to be sent are written in clear into the body of the message.
- **Redundancy check**. A redundancy check (RC) is the simplest to implement and least secure of the security mechanisms outlined in the 3GPP 23.048 specification.

It can only really be used to check that the message was correctly received by the receiving entity. If you select this option:

a) Click **Apply**.

a) Select the **KID** property sheet.

b) In the **RC Security** area of the window, select the **Algorithm** to use to calculate the redundancy checksum.

c) If you select the **DES** or **Triple DES** algorithm, you must enter the **Key** to use.

The receiving entity calculates a checksum based on the contents of the message and compares it with the checksum value sent in the message. If the two match, the receiving entity goes on to process the contents of the message. If a discrepancy is detected, the message is rejected. If status reporting is switched on, an error is returned to the sending entity.

- **Cryptographic Checksum.** A cryptographic checksum is a string of bits derived from a secret key, part or all of the message's contents, and possibly additional information (for example, part of the command header). This secret key must be known to both the sending entity and the receiving entity. If you select this option:

  a) Click **Apply**.

  b) Select the **KID** property sheet

  c) Specify the algorithm and key to use in the **CC Security** area of the window, as described in "KID" on page 93.

  The receiving entity authenticates the message by comparing the content of the CC field extracted from the command packet with a value computed internally by the SIM card using the same secret key as the sender.

- **Digital Signature**. A digital signature (DS) enables the message's recipient to verify the authenticity of the information's origin, and also verify that the information is intact. If you choose this option, you must:

  a) Click **Apply**.

  b) Select the **KID** property sheet.

  c) Specify the algorithm and key to use in the **DS Security** area of the window, as described in "KID" on page 93.

Counter

A synchronization counter is used to prevent replay attacks and to "re-synchronize" the OTA platform and SIM card when OTA message transmission fails for whatever reason. Synchronization consists in comparing two values; one stored in the SMS message's command packet, the second stored on the SIM card itself.

The following options are available:

- **No counter available.** Do not use a synchronization counter.
- **Counter available but no replay or sequence checking.** Include a synchronization counter in the message, but the receiving entity does not perform any checks on its value.
- **Process if counter is higher than RE.** Process the message if the value of the synchronization counter in the message is greater than the current value stored on the SIM card.
- **Process if counter is one higher than RE.** Process the message if the value of the synchronization counter in the message is exactly one more than the current value stored on the SIM card.

Refer to the 3GPP 23.048 specification for details.

If you choose to use a synchronization counter, enter the initial value to use in the **Counter value** field.

The maximum possible value of synchronization counter and the method used to format the synchronization counter can be set on the **Expert** property sheet. See “Expert Options” on page 95.

The receiving entity retrieves the value of the synchronization counter from the message and compares it to the value of the synchronization counter in the SIM card. The message is considered to be valid if the counter’s value is greater than the value stored in the SIM card, in which case the local counter is incremented.

Counter Value

Enter the initial value of synchronization counter to use. The default value is zero (0).

Click the **Refresh** button to read the most recently saved value of synchronization counter from the card profile file. For example, if you replace the current value “0” with “999” and click **Refresh**, the value “0” reappears.

Proof of Receipt

The values you specify for the options in this area of the window are written into the second SPI byte of the message’s command header.

Proceed as follows:

1. To specify that no Proof of Receipt (PoR) is required, select **No PoR reply to the Sending Entity (SE)**.
2. To specify that a PoR is always sent, select **PoR required to be sent to the SE**.
3. To specify that a PoR is only to be sent when an error occurs, select **PoR required only when an error has occurred**.

If you request that a PoR is sent, you can choose the security mechanism to apply to it:

- **No security to be applied**
- **Apply a cryptographic check (CC) check only**
- **Apply a digital signature (DS) check**

Note: Depending on the target chosen, not all of the above options may be available.

You can optionally choose whether the PoR is to be ciphered or not.

Finally, you can specify whether the PoR is to be sent using an SMS-DELIVER-REPORT TPDU or an SMS-SUBMIT TPDU.

These options are explained in greater detail in the 3GPP 23.040 specification.

KIc

This property sheet is used to determine the contents of the key certificate (KIc) byte in the command header.

The values on this property sheet are only used if ciphering of the message has been requested. Check the value of the **Ciphering** field on the **SPI** property sheet, described in “SPI” on page 90.

The window is divided into two parts:

- In the top half of the window, specify the ciphering values to use when data encryption is being used. See “Configuring Data Encryption” on page 93 below.
- In the bottom half of the window, specify the settings for the ciphering key: see “Ciphering Key”.

Ciphering Key

Select the algorithm type and mode to use to cipher the message's contents.

Algorithm. The following algorithms are available:

- **Value set in template**. The value is read from the template upon which this profile is based.
- **DES/CBC/None**. DES in CBC mode, with no padding
- **DES/ECB/None**. DES in ECB mode, with no padding
- **TripleDES/CBC/None**. Triple DES in outer-CBC mode, with no padding
- **TripleDES/ECB/None**. Triple DES in ECB mode, with no padding.

Key. Type the key value to use. The key should be:

- 8 bytes (56 bits) long if the algorithm selected is **DES/CBC/None** or **DES/ECB/None**.
- 8 bytes (56 bits), 16 bytes (112 bits), or 24 bytes (168 bits) long if the algorithm selected is **Triple DES/CBC/None** or **Triple DES/ECB/None**.

KID

This property sheet is used to determine the contents of the key identifier (KID) byte in the command header. The window is divided into two parts:

- In the top half of the window, specify the ciphering values to use when data encryption is being used. See “Configuring Data Encryption” on page 93 below.
- In the bottom half of the window, specify the RC, CC, or DS algorithm and key to use: see “Configuring CC, RC, or DS Security Settings” on page 94.

Configuring Data Encryption

Algorithm Type

First, choose the algorithm type:

- An algorithm known to both the sending and receiving entity.
- A Data Encryption Standard (DES) algorithm.
- A proprietary algorithm.

Algorithm

Choose the encryption algorithm. The OTA Profile Manager supports the following encryption algorithms:

- **DES**. DES uses a binary number called a key to encrypt and decrypt sensitive data. In the OTA Profile Manager, you simply type in the entire 192-bit (24 character) key rather than entering each of the three keys individually.
- **Triple DES**, with two or three keys. Triple DES is simply another mode of DES operation. The procedure for encryption is exactly the same as regular DES, but it is repeated three times (hence the name *triple* DES). The data is encrypted with the

first key, decrypted with the second key, and finally encrypted again with the third key, as shown in “Figure 60”below:

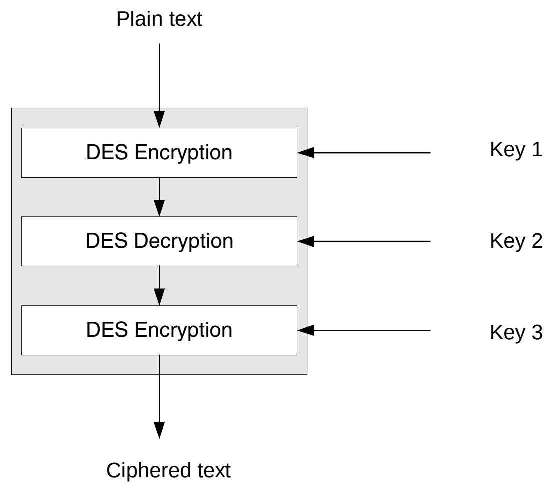


Figure 60 - Triple DES Encryption

In the 3-key variant, the three keys are all different, whereas in the two-key version, **Key 3** is the same as **Key 1**. In both cases, the chaining features of CBC mode are also employed.

Key Set

Choose the key set to use from the key set file. Each key set must contain either 16 characters, that is, 112 bits (when using two 56-bit keys), or 24 characters, that is, 192 bits (when using three 56-bit keys).

Configuring CC, RC, or DS Security Settings

The items in the bottom half of the property sheet depend on the value of the **Security** field on the SPI property sheet (see “Security” on page 90):

- If **Security** is set to **Cryptographic Checksum**, specify a cryptographic checksum in the **CC Security** area:
  a) Select the **Algorithm** to use: XOR4, CRC32, XOR8, DES or Triple DES.
  b) If you select the **DES** or **Triple DES** algorithms, specify the **Key** to use; 8 bytes (56 bits) for DES, 16 bytes (112 bits) or 24 bytes (168 bits) for Triple DES.
- If **Security** is set to the value **Digital Signature**, specify the digital signature to use in the **DS Security** area of the window:
  a) Choose the algorithm: XOR8, DES, or Triple DES.
  b) If you select the **DES** or **Triple DES** algorithms, specify the **Key** to use; 8 bytes (56 bits) for DES, 16 bytes (112 bits) or 24 bytes (168 bits) for Triple DES.
- If **Security** is set to the value **Redundancy check**, specify the algorithm and key to use in the **RC Security** area:
  a) Select the **Algorithm** to use, XOR4 or CRC32.

Expert Options

The options on the **Expert** property sheet are typically parameters that are optional or open to different interpretations in the 3GPP 23.040 or 3GPP 23.048 specifications. Specific values may therefore be necessary to create OTA profiles compatible with different manufacturers' interoperable cards.

The parameters are divided into three sub-property sheets: click on the **Concatenation**, **Command Packet Header** or **Response Packet Header** tabs to the right of the window to access the other property pages.

Concatenation

When concatenation is being used, the first concatenated message contains three elements:

- The command header
- A concatenation header.
- The first part of the secured data, if there is room for it.

Subsequent messages contain only the concatenation header and secured data.

The options on this property sheet set the format of these components.

Concatenation Header Size (IEIa)

The length of Information Element Identifier "A" field to use when formatting messages. Possible values are:

- **Value set in template**, to use the value in the template upon which this card profile is based.
- **One byte (0x00)**, if the concatenation header reference number size is one byte.
- **Two bytes (0x08)**, if the concatenation header reference number size is two bytes.

TPUD Max Length

Defines the maximum possible length of message (TP-User Data field length) that can be generated by the OTA platform's formatting library (UDH plus command header plus secured data). For point-to-point SMS messages, this value must be **140** (bytes). However, you can specify a different value for the maximum length if necessary.

Command Packet Header

CHI expected length

Specify the maximum length of the Command Header Identifier (CHI) field. This field is always null (0 bytes) in SMS point-to-point messages.

CHI value

Defines the expected length of the CHI value in an unformatting process. The most suitable value is "0".

CP synchronization counter maximum

Specify the largest valid value of synchronization counter within a command packet that is supported by the target card.

CP synchronization counter formatter

The default value codes the synchronization counter as a 5-byte unsigned value. For example, the synchronization counter “703696” is coded as “0Ah 0Bh 0Ch 0Dh 00h” (hexadecimal). To use a different format, select **Value set in template** to use the format specified in the template upon which this card profile is based.

Fields included in the RC/CC/DC calculation

For generalized command packets, the RC/CC/DS field is computed from the SPI, KIc, KID, TAR, CNTR, and PCNTR fields, together with additional data and padding. However, part or all of the CPL and CHL fields can also be used in this computation. Therefore, to fully comply with the 3GPP 23.048 specification, the OTA Profile Manager allows you to choose which fields are included in the RC/CC/DS computation.

The default formatting library provides two alternatives:

- Extract data from the SPI, KIc, KID, TAR, CNTR, and PCNTR fields, together with additional data and padding from the input stream.
- Extract the data from the CPL, CHL, SPI, KIc, KID, TAR, CNTR, and PCNTR fields, together with additional data and padding from the input stream.

To use a different format, select **Value set in template** to use the format specified in the template upon which this card profile is based**.**

Data to cipher padder

Determines how to pad data that is to be ciphered. The default formatting library always pads data with zeros.

To use a different format, select **Value set in template** to use the format specified in the template upon which this card profile is based**.**

Response Packet Header

RHI expected length

Specify the maximum length of the Response Header Identifier (RHI) field. The value “0” is the default.

RHI value

Defines the expected length of the RHI value in an unformatting process. The most suitable value is “0”.

RP synchronization counter maximum

Specify the largest valid value of synchronization counter that is supported by the target card within a response packet.

RP synchronization counter formatter

The default value codes the synchronization counter as a 5-byte unsigned value. For example, the synchronization counter “703696” is coded as “0Ah 0Bh 0Ch 0Dh 00h” (hexadecimal). To use a different format, select **Value set in template** to use the format specified in the template upon which this card profile is based.

Fields included in the RC/CC/DS calculation

For generalized response packets, the RC/CC/DS field is computed from the TAR, CNTR, and PCNTR, and status fields, together with additional data and padding. However, part or all of the RPL, RHL, UDHL and RPI UDH fields can also be used in this computation. Therefore, to fully comply with the 3GPP 23.048 specification, the OTA Profile Manager allows you to choose which fields are included in the RC/CC/DS computation.

There are two alternatives:

- Extract data from the TAR, CNTR, and PCNTR fields, together with additional data and padding from the input stream.
- Extract data from the RPL, RHL, TAR, CNTR, and PCNTR fields, together with additional data and padding from the input stream.
- Extract data from the UDHL, RPI UDH, RPL, RHL, TAR, CNTR, and PCNTR fields, together with additional data and padding from the input stream.

To use a different format, select **Value set in template** to use the format specified in the template upon which this card profile is based**.**

Standards and Specifications

- *3GPP TS 23.040 —3rd Generation Partnership Project; Technical Specification Group Terminals; Technical realization of the Short Message Service (SMS) (Releases 5 and 1999)*
- *3GPP TS 23.048 —3rd Generation Partnership Project; Technical Specification Group Terminals; Security mechanisms for the (U)SIM application toolkit; Stage 2, (Release 5)*
- *3GPP TS 03.48 —3rd Generation Partnership Project; Technical Specification Group Terminals; Security mechanisms for the SIM application toolkit; Stage 2; (Release 1999)*
- *3GPP TS 31.101 3rd Generation Partnership Project; Technical Specification Group Terminals; "UICC-Terminal interface; Physical and logical characteristics (Releases 5 and 6)*
- *3GPP TS* 51.011 *3rd Generation Partnership Project; Technical Specification Group Terminals; "Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface" (Release 5)*
- *3GPP TS* 11.11: *3rd Generation Partnership Project; Technical Specification Group Terminals; "Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) Interface" (Release 1999)*
- *3GPP TS 31.111: "Universal Subscriber Identity Module (USIM) Application Toolkit (USAT)" (Releases 5 and 1999)*
- *3GPP TS 11.14: "Specification of the SIM Application Toolkit (SAT) for the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface" (Release 1999)*
- *3GPP TS 31.130—3rd Generation Partnership Project; Technical Specification Group Terminals; (U)SIM Application Programming Interface (API); (U)SIM API for Java Card, (Release 6)*
- *3GPP TS 43.019—3rd Generation Partnership Project; Technical Specification Group Terminals; Subscriber Identity Module Application Programming Interface (SIM API) for Java Card, Stage 2, (Release 5)*
- *3GPP TS 03.19—3rd Generation Partnership Project; Technical Specification Group Terminals; Subscriber Identity Module Application Programming Interface (SIM API) for Java Card, (Release 1999)*
- *ETSI TS 102–221; Technical Specification; Smart cards; UICC-Terminal interface; Physical and logical characteristics (Release 5)*
- *ETSI TS 102–223; Technical Specification; Smart cards; Card Application Toolkit (Release 5)*
- *ETSI TS 102–241; UICC API for Java Card (Release 6)*
- *ISO/IEC 7816-4: 1995: "Identification cards—Integrated circuit(s) cards with contacts— Part 4: Interindustry commands for interchange"*
- *ISO/IEC 7816-6: 1996* "*Identification cards—Integrated circuit(s) cards with contacts— Part 6: Interindustry data elements"*
- *GlobalPlatform Card Specification 2.0.1 from GlobalPlatform, April 7th, 2000.*
- *GlobalPlatform Card Specification 2.1.1 from GlobalPlatform, March, 2003.*
- *Java Card Specifications 2.2.*, Sun Microsystems, June 2002.
- *Java Card Specifications 2.2.1*, Sun Microsystems, October 2003.
- *JSR 185: Java Technology for the Wireless Industry*, Sun Microsystems, July 2003

Web Site Addresses

www.gemalto.com for more information about Gemalto products and services.

www.borland.com to download Eclipse.

www.globalplatform.org to download copies of the GlobalPlatform (Open Platform) specifications.

www.java.sun.com for anything relating to Java.

www.developers.sun.com for introduction to Java (see “Recommended Reading”)

www.opencard.org for information about the Open Card Framework.

www.etsi.org for information about European Telecommunications Standards (3GPP).

Recommended Reading

For more information about Java applet development for smart cards, see:

- *An Introduction to Java Card Technology*, parts 1–3, from www.developers.sun.com/techtopics/mobility/javacard/articles/javacard1/
(for parts two and three the link is the same but with 2 or 3 at the end, respectively)
- *Application Programming Notes*, *Java Card Platform Version 2.2.1* from Sun Microsystems, October 2003
- *Open Card Framework 1.2 Programmer’s Guide*, available from: http://www.opencard.org.

Abbreviations

3GPP	3rd Generation Partnership Project
AID	Application Identifier
APDU	Application Protocol Data Unit
API	Application Programming Interface
ATF	APDU Trace Format
ATR	Answer To Reset
BIP	Bearer Independent Protocol
CAD	Card Acceptance Device
CAP	Card Applet Package
CAT	Card Application Toolkit
CBC	Cipher Block Chaining
CCAT	CDMA Application Toolkit
CLA	Class byte
CLDC	Connected Limited Device Configuration.
CPLC	Card Production Life Cycle
DES	Data Encryption Standard
ECB	Electronic Code Book
ETSI	European Telecommunications Standards Institute
EXP	Java Card Export file
FSD	Full Serialized Data
GSM	Global System for Mobile communications
HTML	HyperText Markup Language,
HTTP	HyperText Transfer Protocol
ICV	Initial Chaining Vector
IDE	Integrated Development Environment
INS	Instruction byte
ISO	International Standards Organization
J2ME	Java 2 Platform Micro Edition
JAR	Java Archive file
JCA	Java Card Assembler

JCRE	Java Card Runtime Environment
JCVM	Java Card Virtual Machine
JDK	Java Development Kit
JSR	Java Specification Request
JVM	Java Virtual Machine
Lc	Data length
Le	Expected length of data to be returned
MAC	Message Authentication Code
ME	Mobile Equipment
MIDP	Mobile Information Device Profile
OCF	OpenCard Framework
OMA	Open Mobile Alliance
OP	Open Platform
OTA	Over The Air
PC/SC	Personal Computer/Smart Card
PIX	Proprietary Identifier Extension
PK	Public Key
PSK-TLS	Pre-Shared Key-Transport Layer Security
RID	Registered Identifier
R-UIM	Removable User Identity Module
SAP	Simulator Applet Package (card simulator-proprietary load format)
SAT	SIM Application Toolkit
SATSA	Security And Trust Services API
SCWS	Smart Card Web Server
SIM	Subscriber Identity Module
TCP/IP	Transmission Control Protocol/internet Protocol
TLS	Transport Layer Security
TPDU	Transport Protocol Data Unit
TP–UD	TP User Data
UICC	Universal Integrated Circuit Card
URI	Uniform Resource Identifiers
URL	Uniform Resource Locators
USAT	USIM Application Toolkit
USIM	Universal Subscriber Identity Module
VM	Virtual Machine
WTK	Wireless Toolkit

Glossary

Applet	In Java Card terminology, a Java Card applet is an independent Java application loaded into a Java Card.
Application identifier	A string of between 5 and 16 bytes that identifies a package or an application in a card and which corresponds to the naming scheme defined in ISO7816-5. It may contain a registered application provider number. If it contains no application provider number, then this identification may be ambiguous.
Application Protocol Data Units (APDU)	Standard communication messaging protocol between a card acceptance device and a smart card.
Application Provider	The entity that owns an application and is responsible for the application’s behavior.
ATF file	A Gemlato APDU Trace Format script file.
ATR	When a card is inserted into a card reader, it stimulates a contact, which provokes the terminal to reset itself by sending a reset signal to the card. The card’s response is called an Answer To Reset (ATR). The ATR is described in two standards: *ISO 7816-3: Electronic signals and transmission protocols*, which defines the two possible low-level communication (or transport) protocols between the terminal and the card. It is strongly advised to refer to this standard for details. *ISO 7816-4: Interindustry commands for interchange*, which defines a set of standard commands for smart cards, as well as a hierarchical file system structure for cards. These commands are the basis of most existing card protocols.
Bytecode	A VM instruction code as a sequence of binary bytes.
CAP files	A file format that can be loaded into a card. CAP files are generated by Simulation Suite V2.6.
Card Application Toolkit (CAT)	A set of generic commands and procedures for use by the UICC, irrespective of the access technology of the network (for example, USAT for 3G, or CCAT for CDMA).
CCAT	CDMA Card Application Toolkit. A set of commands and procedures for use during the network operation phase of CDMA.
CAT	See Card Application Toolkit.
Card Simulator	A software environment that simulates a Java Card context. Used for testing applet functionality before loading the applet into a card.
Class files	A compiled Java code file.
Client application	An application that requests services from a an applet in the Java Card environment.

Connected limited device configuration (CLDC)	Defines the base set of application programming interfaces and a virtual machine for resource-constrained devices like mobile phones, pagers, and mainstream personal digital assistants
Conversion	A post-compilation step necessary to convert Java Class bytes into a form (.CAP files) understood by the card. This is carried out automatically by Simulation Suite V2.6 when the project is built.
Converter	A utility provided by Sun to convert Java Class files into a form understood by the card.
JAR file	Java Archive file. A compressed file containing Java Class files, or CAP load files. Used by the GxpLoader, for example.
Java Card-GP applet	An applet that is fully compliant with the Java Card and GlobalPlatform specifications.
Java-GP Card	Smart cards that are fully compliant with both the Java Card and GlobalPlatform specifications.
JCA file	A Java Card Assembler file, generated by the GxpConverter or Sun’s Converter tool.
JCardManager	A Simulation Suite V2.6 tool that simulates a client application, enabling a user to exchange commands with an applet installed in either a card or the card simulator.
Key index	Identifies a key within a key set.
Key set	A set of encryption keys used to authenticate the owner.
Key version	A number between 0 and 127 identifying the key version to use.
Load file	The physical data files that are uploaded to Java-GP cards to modify the card’s contents.
MAC (Message Authentication Code)	A symmetric cryptographic transformation of data that provides data origin authentication and data integrity.
MIDlet	A MIDlet is an application written for MIDP. MIDlet applications are subclasses of the javax.microedition.midlet.MIDlet class that is defined by MIDP.
Mobile Information Device Profile (MIDP)	Mobile Information Device Profile. Set of Java APIs that is generally implemented on the Connected Limited Device Configuration (CLDC). It provides a basic J2ME application runtime environment targeted at mobile information devices, such as mobile phones and two-way pagers. The MIDP specification addresses issues such as user interface, persistent storage, networking, and application model.
OTA interpreter	The OTA Interpreter tool displays an “interpreted” view of a sequence of OTA data bytes. Details of the interpretation are displayed for envelope, command packet header, and response packet message parts, according to a specific OTA profile and the relevant 3GPP TS 23.048 specifications

Package	A Java term to describe a collection of related classes and interfaces. A package can contain, for example several applets.
PC/SC	The PC/SC (Personal Computer/Smart Card) Workgroup was formed in May 1996 to resolve interoperability issues between PCs and smart cards. Members include Gemalto, IBM, Sun Microsystems, Toshiba, Groupe Bull, Hewlett-Packard, Microsoft, Schlumberger, and Siemens Nixdorf.
Project	A collection of one or more applets, together with the necessary system classes, with or without libraries. Also used to describe a Eclipse IDE project.
RID	The first five bytes of an AID, registered with ISO, indicating the Application Provider.
SCR file	A Sun script file.
Scripts	A list of commands. Gemalto scripts (.atf) or Sun scripts (.scr) can be exchanged with a card or the card simulator.
SIM Application Toolkit (SAT)	A set of commands that defines how a card should interact with the outside world and extends the communication protocol between the card and the handset. With SAT, the card has a *proactive* role in the handset (this means that the SIM initiates commands independently of the handset and the network). In 2G networks, SIM Application Toolkit (SAT) was defined in the GSM 11.14 standard.
SIM Toolkit	A set of commands and procedures for use during the network operation phase of GSM, in addition to those defined in TS 11.11.
Target	The entity in which a Java Card applet or package is eventually installed. The target can be either the card simulator or a specific type of card in a card reader.
Trace	A log that shows the results of exchanges between the reader and the card or card simulator. Traces can be saved in a trace file.
Universal Integrated Circuit Card (UICC)	A physically secure device, an IC card (or “smart card”), that can be inserted and removed from the terminal equipment. It may contain one or more applications. One of the applications may be a USIM.
USIM Toolkit	Provides mechanisms which allow applications, existing in the UICC, to interact and operate with any ME which supports the specific mechanism(s) required by the application.These mechanisms are dependent upon the commands and protocols relevant to USAT in 3GPP TS 31.101.
Wear Leveling	Wear leveling is a process of distributing set of writes across more flash pages/sectors, so as to ensure that the flash memory is used in a uniform fashion. This process decreases the total wear on the memory, thereby increasing the lifetime of the card.

Wireless Toolkit	The Sun Java Wireless Toolkit (formerly known as J2ME Wireless Toolkit) is a set of tools for creating Java applications that run on devices compliant with the JSR185 specification. It consists of build tools, utilities, and a device emulator.
Wizard	A predefined project builder provided by Simulation Suite V2.6 for the Eclipse IDEs to help you specify project-related information. The wizards generate ready-to-compile skeleton Java code, to which the developer need only add the applet or client application’s functionality.