Automated Testing Environment for SIM Application Toolkit

Cristian Radu
Automated Testing Environment for SIM Application Toolkit

Dr. Cristian Radu
Integri n.v.,
Leuvensesteenweg 325,
Zaventem, BELGIUM
e-mail : cradu@integri.com
URL: http://www.integri.com
Outline of the talk

- Introduction
- SIM/ME Software Certification
- Test management using INQSmart
- Testing SIM-ME interface
- Testing SIM chip cards
- Testing ME
Introduction

Scope of the paper: Test environment and test procedures used for evaluating the conformity of the software implementation in a SIM chip card/ME with respect to their functional specifications.

The proposed test environments are built using hardware/software components from the INQSmart toolbox, which is a testing technology developed by INTEGRI.

Functional specifications used for the purpose of test definition phase are elaborated by ETSI:

- GSM 11.11 -- SIM-ME interface
- GSM 11.14 -- SIM Application Toolkit

Test specifications elaborated by ETSI:

- GSM 11.17 -- Test Specification of the SIM chip card
- GSM 11.10-4 -- SIM Application Toolkit conformance specification
SIM-ME (MS) working environment

- MS
- SIM Chip Card
- ME
- GSM 11.11
- GSM 11.14
- GSM Network

EuroSTAR 2000  December 6, 2000  Copenhagen
Issues when setting up the Test Environment

- User interface not very relevant --> record and playback techniques not ideal
- Interfaces play a crucial role in the processing
- All units are finite state machines
- Security complicates testing
Test Management using INQSmart

Overview of the GSM 11.17 Test List

![Test List](image-url)
Node refinement in the GSM 11.17 Test List

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disabled</td>
<td>False</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>Path</td>
<td>SIM.FN.Description</td>
</tr>
<tr>
<td>Value</td>
<td>The tests in this subclause ensure that the IUT conforms to the functional specification for all the commands and their respective responses.</td>
</tr>
</tbody>
</table>
Set of conformity requirements identified for a function
Name of the test:

T1: Test that reading the first record from EF_ADN is not possible without CHV1.

Description of the test case:

This test exercises the conformity requirement CR4.

The internal status of the SIM card is set up such that:

- EF_ADN as the current EF, \texttt{currEF = EF_ADN}.
- There is no current record selected, \texttt{recPointer = 0}.
- EF_ADN is a linear fixed EF, \texttt{isLinearorCyclic = True}.
- There was no correct submission of CHV1 preceding the test, \texttt{readAC = False}.

T1 is an invalid test, which tries to trap the SIM to do what it should not do, i.e., allowing reading without a prior verification of the READ access condition, which consists of submitting CHV1 (Cardholder Verification Code).
<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim</td>
<td>Dim FIDDF as String ; Dim FIDEF as String ; Dim Buffer_CC as String</td>
</tr>
<tr>
<td>Reset</td>
<td>Call ICC_Reset</td>
</tr>
<tr>
<td>Initialise DF</td>
<td>FIDDF = &quot;7F10&quot;</td>
</tr>
<tr>
<td>Select DF</td>
<td>Call ICC_GenerateTemplate(&quot;SELECT&quot;) ; Call T_Sel(&quot;Command&quot;)</td>
</tr>
<tr>
<td>Selection DF eval</td>
<td>Buffer_CC= T_Get(&quot;Response_CC&quot;) ; Buffer_SW1=Str_Left(B1, 10)</td>
</tr>
<tr>
<td>Initialise EF</td>
<td>FIDEF = &quot;6F40&quot;</td>
</tr>
<tr>
<td>Select EF</td>
<td>Call ICC_GenerateTemplate(&quot;SELECT&quot;) ; Call T_Sel(&quot;Command&quot;)</td>
</tr>
<tr>
<td>Selection EF eval</td>
<td>Buffer_CC= T_Get(&quot;Response_CC&quot;) ; Buffer_SW1=Str_Left(B1, 10)</td>
</tr>
<tr>
<td>GetResponse</td>
<td>Call ICC_GenerateTemplate(&quot;GET_RESPONSE&quot;) ; Buffer_SW1=Str_Left(B1, 10)</td>
</tr>
<tr>
<td>Prepare compar...</td>
<td>Buffer_Data = T_Get(&quot;Response.Data&quot;) ; Buffer_FType = Micro</td>
</tr>
<tr>
<td>Comparison</td>
<td>if Buffer_FType &lt;&gt; &quot;04&quot; then ; Err.Raise TEST_ERR, &quot;7th I</td>
</tr>
<tr>
<td>Result Last Exec...</td>
<td>Test Ended OK ; Date/Time = 1999-09-14 12:09:40.520</td>
</tr>
</tbody>
</table>

```
Dim FIDDF as String
Dim FIDEF as String
Dim Buffer_CC as String
Dim Buffer_SW1 as String
Dim Buffer_SW2 as String
Dim Buffer_Data as String
Dim Buffer_FType as String
```
Testing SIM-ME interface

INQSmart in Logger mode for SIM-ME interface
Configuration

- INQSmart software package configured in the Logger mode
- PC running the INQSmart software package
- ME in which the probe of the hardware adapter can be inserted
- SIM chip card
- Hardware adapter (HWA) of type STAR 1150 delivered together with a probe connected to the HWA through a multiple-wire cable.

In the Logger mode data on the SIM-ME interface are sampled in order to:

- analyse the ATR
- analyse the time intervals between the transitions
- convert physical transitions in bytes
- convert streams of bytes in APDU
Testing SIM chip cards

Master -- slave relationship

ME (Master)

I/O

Command/Response

Clock

Reset

Power

GND

SIM (Slave)

I/O

Reset

ATR

Adapt transmission according to ATR

Command

Response

Process command

Elaborate response

EuroSTAR 2000

December 6, 2000

Copenhagen
Modelling

- unit under test is modelled = simulation program based on functional specifications
- during testing model is synchronised with the unit under test
- any discrepancy between model and unit under test = error
- allows full automation of tests
- allows small test scripts
Step 1: Decomposing the System
Step 2: Implementing the Interface
Step 3: Modelling
Step 4: Test Mechanism

TEST ENVIRONMENT

Test Scripts → THALES

Card Reader

Card Model (Data + Processing)

INTERFACE

BlackBox

Robert Schumann expires 01/00

TEST ENVIRONMENT

EuroSTAR 2000
December 6, 2000
Copenhagen
Test environment for SIM chip cards

INQSmart in Test Tool Mode for SIM testing
Configuration

- INQSmart software package configured in the Test Tool mode
- PC running the INQSmart software package
- Card reader in which the SIM card under test can be inserted either directly (ID-1 SIM) or using an adequate adapter (Plug-in SIM)
- SIM chip card, which is submitted to the software certification procedure

Session Components

- GSM 11.17 Test Library
- Definitions Tree
- Card Tree
- Log
- DLL
SIM behaves as a pseudo-master demanding some services from the terminal. Therefore, from the point of view of testing, the SIM must be replaced with an emulator, flexible enough to program as many tests as required (GSM 11.10-4).
Configuration

- INQSmart software package configured in the Emulator mode
- PC running the INQSmart software package
- Hardware adapter (HWA) of type STAR 1150 delivered together with a probe connected to the HWA through a multiple-wire cable.
- ME in which the probe of the hardware adapter can be inserted

Session Components

- GSM 11.10-4 Test Library
- Definitions Tree
- Card Tree
- Log
- DLL
Definitions Tree data structure contains:
- the syntax of the command/response pairs
- the completion codes (the status words SW1, SW2) of the commands supported by the emulated card.
- the message identification patterns of the commands
- the templates of the file headers in the emulated card as well as the templates of its Elementary Files.
Card Tree data structure contains the main resources of the emulated card. These resources consists of:

- the file structure of the emulated card,
- a set of temporary variables used during the processing
- the I/O buffer, storing the command received from the Test Manager and the response computed by the emulated card.
GSM 11.10-4 Test List

To verify that the ME displays the text contained in the DISPLAY TEXT proactive SIM command, and returns a successful result in the TERMINAL RESPONSE command send to the SIM.
GSM 11.10 -4 Test Library

![GSM 11.10-4 Test Library Diagram]

- Test Library
  - Dim
  - Set TimeOut
  - Initialisation of SIM Application Toolkit Enabled SIM by SIM
  - Proactive SIM Commands
  - DISPLAY TEXT
    - Prerequisite
    - VERIFY_CHV
      - Put DISPLAY TEXT proactive command in list
    - TERMINAL_RESPONSE
      - Check if TERMINAL RESPONSE ended by
  - Buffer

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created On</td>
<td>00/09/18 6:13:58 I</td>
</tr>
<tr>
<td>Name</td>
<td>DISPLAY TEXT</td>
</tr>
<tr>
<td>Path</td>
<td>Test Library.Proact</td>
</tr>
<tr>
<td>Result Last Exec.</td>
<td>Test Ended NOK ;</td>
</tr>
<tr>
<td>Result Tree</td>
<td>...</td>
</tr>
</tbody>
</table>

Test Ended NOK : Date/Time = 2000-09-19 12:24:48.346
Description = Error encountered in event
<Test_TERMINAL_RESPONSE> : Err.Source="Test_TERMINAL_RESPONSE" ; Err.Description="ME currently unable to process command"
Conclusions

Testing the correctness of software implementations in SIM chip cards and ME is difficult.

The use of a SIM card emulator can facilitate the SIM testing, because the test operator has a reliable reference for the results computed by the card under test.

The SIM card emulator can be used both for the test of the SIM card but also of the ME requiring services from this card.

INQSmart is a suitable testing tool for SIM cards and ME.

For further inside on INQSmart visit the URL: http://www.integri.com
Automated Testing Environment for SIM Application Toolkit

Dr. Cristian RADU
Integri N.V.
Leuvensesteenweg 325,
B-1932, Zaventem, Belgium
Tel: +32-2- 713 08 48
Fax: +32-2-712 07 67
email: cradu@integri.com
URL: http://www.integri.com

Abstract
The SIM Application Toolkit provides mechanisms that allow applications existing in the SIM pro-active chip card to interact and operate with any ME (e.g., GSM handset), which supports the specific mechanisms required by the application. Even that ETSI elaborated norms covering the test definition phase for both SIM pro-active chip card testing and ME testing very little literature describes the corresponding testing environments. A test environment encompasses the necessary hardware/software configuration that is needed to allow the test preparation and test execution phases. This paper proposes automated test environments allowing for efficient testing of SIM pro-active chip cards and ME. The implementation of the proposed environments is carried out using specific hardware/software components from the INQSmart family, promoted by INTEGRI.

1 Introduction

In the GSM scenario, a Mobile Station (MS) is the terminal of the subscriber when using the GSM network infrastructure. The MS consists of a Mobile Equipment (ME) and a Subscriber Identity Module (SIM) chip card. This working environment is presented in Figure 1.

![Figure 1 Working environment for the GSM scenario](image)

The SIM is primarily used in the GSM network operations, supporting the subscription and security related procedures. In this case the SIM interfaces with the ME, with which command/response pairs are exchanged, according to GSM 11.11. This standard defines that the ME communicates to the SIM using the T=0 protocol, which is specified in ISO/IEC 7816-3. The ME is always the “master” and initiates commands to the SIM, which elaborates and sends back responses.

In the scenario described above, there is no mechanism for the SIM to initiate a communication with the ME. This limits the possibility of introducing new SIM features requiring the support of the ME, as the ME should know in advance what actions it should take. The SIM Application Toolkit provides mechanisms allowing applications existing in the SIM to interact and operate with any ME that supports the specific
mechanisms required by the application. These mechanisms are specified in GSM 11.14, and they are dependent upon the commands and protocols relevant to SIM Application Toolkit in GSM11.11. The proactive SIM service provides a mechanism, which stays within the protocol T=0, allowing the SIM to say that it has some information to send to the ME. This information conveys a command to be executed by the ME, which will also elaborate the response for the SIM to inform it about the completion of this command.

From the viewpoint of software testing of SIM chip card and ME implementations, ETSI elaborated two important norms, namely:

- the Test Specification of the SIM chip card in GSM 11.17;
- the SIM Application Toolkit conformance specification in GSM 11.10-4.

The standard GSM11.17 specifies the test cases for the SIM chip card, according to the specification of the SIM-ME interface presented in GSM 11.11.

The standard GSM 11.10-4 refers to the test cases concerning the ME, according to the specification of the SIM Application Toolkit mechanisms described in GSM 11.14.

Thus, a very important step towards the test analysis of the software certification procedures for SIM chip cards and ME was achieved. However, very little literature detail test environments which allow the implementation of these software certification procedures. A test environment specifies the necessary hardware/software components that are needed to allow the test preparation and test execution phases of a software certification procedure.

This paper describes appropriate test environments for the SIM chip card and ME testing. These environments are built using hardware/software components from the INQSmart toolbox, which is a testing technology elaborated by INTEGRI. It is described a separate test environment for each of the testing activities enumerated below:

- Recording and analysing the transaction flow on the SIM-ME interface, determined by the execution of an application in the SIM chip card. For this task the INQSmart test tool is configured in the Logger mode.
- Testing the SIM chip card according to the test definition phase specified in GSM11.17. For this purpose the INQSmart test tool is configured in the Test Tool mode.
- Testing the ME according to the test definition phase specified in GSM 11.10-4. In this case the INQSmart test tool is configured in the Emulator mode.

The remainder of the paper is organised as follows. Section 2 outlines the features of the SIM/ME software certification activity, and determines its place in the software developing process for SIM chip cards and ME. Section 3 describes the test management of these software certification procedures using the facilities provided by the INQSmart utility. Section 4 describes the test environment for recording and analysing a transaction flow on the SIM-ME interface. Section 5 details the test environment for SIM chip card testing. Section 6 proposes a suitable test environment for ME testing. Finally, our conclusions are presented.

2 SIM/ME Software Certification

Let us considering the V-diagram of a software development project in the case of SIM chip cards and ME, as presented in Figure 2.

In this case the Definition Phase was performed by ETSI. The functional specifications for both the SIM and the ME are standardised in the GSM11.11 and GSM11.14 norms, and are publicly available for software vendors to implement them.

After the analysis of these functional specifications, the vendor performs the Design Phase according to its own understanding of the functional specifications and appropriate to its technological possibilities. This phase elaborates the design specifications to be implemented in the Construction Phase. The implementation is then submitted to Factory Testing, which assesses the correctness of the resulting implementation with respect to its design specification.
Finally *Acceptance Testing* is performed by the GSM network operator, which will accept or reject the software product offered by the vendor.

It could happen that after the acceptance phase of a software product the GSM network operator can face a difficult situation. His clients using the software product during the operation of their GSM handsets complain about the fact that it is inconsistent with its functional specification claimed by the GSM network operator. It is in fact the situation when *independent testing* is needed. That is way the middle layer between the operator and the vendor was added in Figure 2.

![Figure 2: Testing of the SIM chip card and ME](image)

The three phases that are pursued during the independent testing activity are itemised below:

- **Test Analysis**: ETSI, through its specialised groups, has analysed the functional specifications of the SIM chip card and ME, respectively GSM 11.11 and GSM 11.14. Each function is split in a set of *conformance requirements*. For each conformance requirement at least one test is defined. The complete lists of test cases for SIM and ME were derived in two other norms, as follows:
  - Test Specification of the SIM chip card, which is presented in GSM 11.17. This standard specifies the test cases for the SIM chip card, according to the specification of the SIM-ME interface presented in GSM 11.11.
  - SIM Application Toolkit conformance specification, which is presented in GSM 11.10-4. This standard refers to the test cases concerning the ME, according to the specification of the mechanisms described in GSM 11.14.

Therefore, one can say that the testing context in the case of SIM and ME is a *software certification*, since the specification of tests is imposed by ETSI. All actual implementations of SIM chip cards and ME have to be validated against these certification procedures in order to obtain the qualifier “GSM 11.17 certified” for SIM cards and “GSM 11.10-4 certified” for ME.

INQSmart allows defining all the test cases in a hierarchical manner. For each test or group of tests the test designer can indicate the reference to the functional and testing specifications, a short description of the test, its version number, and other information customised by the test designer according to her needs. At the end of this phase the *Test List* is completed, which specifies in fact all the test procedures needed to be implemented in the Test Preparation stage. INQSmart can be customised to include GSM11.17 Test List only, GSM11.10-4 Test List only, or both.

- **Test Preparation**: Based on the GSM11.17 Test List and GSM11.10-4 Test List, INTEGRI has programmed the actual *test scripts* for each test procedure defined in each of these two Test Lists. Test scripts are written in an easy-to-learn Basic-like language. The collection of all the test scripts
forms the Test Library. The INQSmart testing environment can be customised to include the GSM11.17 Test Library only, GSM11.10-4 Test Library only, or both. The test designer also indicates the "many-to-many" links between a subset of test cases from the Test List and the corresponding test scripts in the Test Library. This facility allows automation of test reporting after the test execution phase.

- **Test Execution**: During the test execution phase, INQSmart is configured appropriately to the corresponding test environment. INQSmart runs each test procedure indicated in the Test Library provided with the test environment. All the command/response pairs exchanged between INQSmart and the SIM chip card under test are logged. If the model of the SIM chip card is included in the test environment, the command/response pairs exchanged between the Test Manager and the model of the SIM chip card are also logged. An indication about the finality of a test is also logged, and in case the test failed a possible diagnosis of the defect is provided to the test operator. Moreover, based on the links established between the Test List and the Test Library, there exists the possibility of automated reporting of the execution results directly in the Test List, where the test operator has all the context characterising a test.

INQSmart stores all test information in a repository and defines a number of standard test reports and statistics.

### 3 Test Management using INQSmart

In this section we present the way a Test List and a Test Library can be organised using the test management facilities offered by the INQSmart utility. For the purpose of the paper, we exemplify the test management process with respect to GSM 11.17 Test List and GSM 11.17 Test Library.

The GSM 11.17 Test List is organised as a tree data structure, with a general view as described in Figure 3. It consists of a primer root node, which is named corresponding to the unit under test. In this case the root node of the GSM 11.17 Test List is SIM, from the Subscriber Identity Module card (SIM card) used in the GSM application. The primer node can have several properties, among which the most important are:

- general description of the test project;
- overall comments;
- references used in the developing of the test project;
- time estimation to prepare the tests contained in the Test List;
- time estimation for running the proposed tests.

Each property is referred to with a name and it has a certain content. In the particular example presented in Figure 3, only the reference property is explicitly defined. There are two items of references, the names of which are Ref0 and Ref1, the first pointing to the Test Specification of the SIM chip card (GSM 11.17), and the second pointing to the functional specification of the SIM chip card (GSM11.11). The other properties, e.g., description, comments, and time estimations, can be considered as being non-defined for the root node. An alternative possibility is to define them with an "empty" embodiment.

The root node also has a number of children-nodes organised as folders, each containing a separate Test Group, which corresponds to a chapter from the functional specification GSM 11.11. In this example the root node contains 6 children-nodes:

- **LOG** - *Logical model*, specified in Chapter 6;
- **SEC** - *Security features*, specified in Chapter 7;
- **FN** - *Description of the functions*, specified in Chapter 8;
- **CMD** - *Description of the commands*, specified in Chapter 9;
- **CEF** - *Contents of the Elementary Files (EF)*, specified in Chapter 10;
- **APP** - *Application protocol*, specified in Chapter 11.
Each of the children-nodes can inherit the properties of the root-node. In this case the name and the content of a property from the root-node is visible from the level of a child-node. If a child-node wants to modify a property inherited from the root-node, a property with the same name as the property at the level of the root-node is defined at the level of the child-node, the content of which is modified according to the particularities needed at this level.

A child-node corresponding to a chapter of the functional specification can be further refined in smaller children-nodes corresponding to sections in that chapter. Moreover, the child-node can have properties defined at its own level, which are not visible in the scope of the root-node.

For example, the child-node FN – Description of the functions, corresponding to Chapter 8 of the GSM11.11 specification can be further divided in a number of 17 smaller children-nodes, each corresponding to a separate section in Chapter 8, describing one function supported by the SIM chip card. The process of refining nodes is presented in Figure 4.
As one can see, the node FN has a new property named Description, which is defined at its own level and cannot be seen at the level of the root-node SIM. However, the reference properties Ref0 and Ref1 defined at the level of the root-node SIM, are inherited by FN and are visible at its level.

The further refining of each child-node beneath the FN level can lead to considering the set of test conditions/conformity requirements (CR) identified for a SIM function, which is suggestively presented in Figure 5. The node READ RECORD, which is a child node of FN, defines its own properties:

- **Sec0** -- indication to the section in the test specification GSM11.17 where the READ RECORD function is exercised.
- **Sec1** -- indication to the section of the functional specifications GSM 11.11 were the READ RECORD function is described.
- **Definition** -- a definition of the kind of testing that is performed.
- **Purpose** -- a short presentation of the testing goal with respect to this function.

At the level of the READ RECORD node all the properties inherited from all the ancestor-nodes are also visible.

![Figure 5](image)

**Figure 5** Set of test conditions/conformity requirements identified for a SIM function

The lowest level of nodes in the GSM 11.17 Test List are the nodes defining the conformity requirements (CR). For each node representing a conformity requirement one can find the list of all the test cases defined to cover that conformity requirement. These are actually the leaves of the Test List tree. Each leaf consists of the name of the test and a short description of what the test is doing. The information about test case is divided in the name of the test case and the description of the test case:
Name of the test:
T1: Test that reading the first record from EF_ADN is not possible without CHV1.

Description of the test case:
This test exercises the conformity requirement CR4.
The internal status of the SIM card is set up such that:
- EF_ADN as the current EF, currEF = EF_ADN.
- There is no current record selected, recPointer = 0.
- EF_ADN is a linear fixed EF, isLinearorCyclic = True.
- There was no correct submission of CHV1 preceding the test, readAC = False.

T1 is an invalid test, which tries to trap the SIM to do what it should not do, i.e., allowing reading without a prior verification of the READ access condition, which consists of submitting CHV1 (Cardholder Verification Code).

A test case from the GSM 11.17 Test List is implemented as a set of Test Script nodes in the GSM 11.17 Test Library, with at least one element in the set. A Test Script node contains a number of Test Code nodes, Test Query nodes, Test Grid nodes, and test properties. Each Test Code node contains a number of code lines written in the Integri Script Language.

When the Test Manager engine retrieves the whole set of test code lines, test query lines, test properties, and test grids for a given set of Test Script nodes that form the implementation of a test case, it proceeds as follows:
- The code path in the Test Library tree is formed from the highest root-node level, to the Test Script nodes, which actually contains the Test Code nodes implementing a test.
- Each node within the code path is a Folder or Group node, except for the last node which is the actual Test Script node, containing the Test Code nodes.

Let us look in Figure 6 to an example from the GSM 11.17 Test Library to illustrate the algorithm of establishing the path for retrieving the whole set of test code lines referring to the implementation of a test case.

In this example the name of the Test Script node is 6F40 identifies EF under DF. This test script corresponds to a test case that checks whether the file type identifier (FTI) of the elementary file with the file identifier FID = 6F40h is FTI = 04h, corresponding to an elementary file located into a dedicated file. In this case, the code path is formed from the group nodes with the following names LOG, FID, FTI, EF under DF_TELECOM, and the Test Script node with the name 6F40 identifies EF under DF.

- The nodes that contain the code lines that effectively form the implementation of a test case are registered in an implementation list. Each time a Test Code node, a Test Query node, a Test Property node, or a Test Grid node is encountered under one of the Group nodes or Test Script node found on the path, it is added in the implementation list of the test:
  - in case the node has a name which it is not contained in the implementation list, the node is appended to the end of the list. This is the principle of generalisation through inheritance applied to the code path.
  - in case the node has a name which it is already contained in the list, the node overwrites the existing node in the list. This is the principle of specialisation through overwriting applied to the code path.
Coming back to the GSM11.17 example, one can notice that the implementation list is formed as follows:

- **(Generalisation through inheritance)** Add the Test Code node with name Dim from the level of the Group node with name FID.

- **(Generalisation through inheritance)** Add all the Test Code nodes with the names Reset, Initialise DF, Select DF, Selection DF evaluation, Initialise EF, Select EF, Selection EF evaluation, Get Response, Prepare comparison, and Comparison from the level of the Group node with name FTI.

- **(Specialisation through overwriting)** Replace in the implementation list the Test Code nodes with the names Initialise DF and Comparison, which were added at the previous step, with the Test Code nodes with the same names found under the Group node EF under DF-TELECOM. These nodes contain specialised code for the case when the DF is the DF-TELECOM file and replace the generic code found one level above.

- **(Specialisation through overwriting)** Replace in the implementation list the Test Code node with the name Initialise EF, which was added two steps before, with the Test Code node with the same name found under the Test Script node 6F40 identifies EF under DF-TELECOM. This node contains specialised code for the case when the EF has the concrete FID=6F40h, which replaces the generic case found two levels above that refers to any elementary file.

The order of execution of the nodes will be that in the implementation list. For the specific example we were presenting, the resulting implementation list is visualised in the up-right part of the Test Library window.

The main advantage of using the inheritance and overwriting principles is that the code written for tests is more economical. This fact has direct implications on the productivity of writing tests but also on the maintainability of tests.

Considering again the example presented above, one can see the effectiveness of applying these two techniques. In fact "almost" the same code has to be run for elementary files located under different dedicated files, and having different file ID (FID) numbers.

As it can be noticed, the differences in setting up the dedicated file DF under which an EF is searched are reflected in a first specialisation performed at the level of the Group node EF under DF-TELECOM. The Test Code node with the name Initialise DF passes the ID of the DF under which the EF is searched with a SELECT command. The Test Code node with the name Comparison makes a
distinction of what does it mean a failed selection. The treatment is different when the DF is compulsory to be in the file structure of the card -- it is the case of the DF-GSM -- or when the DF is optional in the file structure of the card -- it is the case of DF-TELECOM.

The second level of specialisation is determined by the differentiation in the file identifier FID for each EF. The Test Code node with the name Initialise EF passes the FID of the EF, the type identifier of which is verified.

If the inheritance and overwriting would not work, the same sequence of code found under the Group nodes LOG, FID, and FTI should be copied and repeated many times for each different EF. This would make the length of the Test Library bigger, and the maintainability more difficult. Imagine that one line of code has to be modified. In this case the test developer has to operate the modification in each copy describing a test for a specific FID. However, if inheritance and overwriting are working, one has to operate the modification only once in the abstract super-class, modification that will be sensed through inheritance by all the test scripts specialised from the abstract super-class.

4 INQSmart in Logger Mode for ME-SIM interface

In order to set up a test environment using INQSmart in the Logger (or Sniffer) mode, one needs the following items (see Figure 7):

- INQSmart software package configured in the Logger mode.
- PC running the INQSmart software package.
- ME in which the probe of the hardware adapter can be inserted.
- SIM chip card, which conforms to GSM 11.11 and GSM 11.14. It can store a certain operator-specific SIM application.
- Hardware adapter (HWA) of type STAR 1150 delivered together with a probe connected to the HWA through a multiple-wire cable. One side of the probe is inserted into the ME, while at the other side the SIM chip card is inserted into the probe. The hardware adapter is connected to the PC through a TCP/IP connection.

Figure 7 INQSmart in Logger Mode

The Hardware Adapter captures the electrical signals exchanged at the interface between the SIM card and the ME. It samples every 50 nanoseconds the electrical signals and all this sampled data is
downloaded from the Hardware Adapter to the INQSmart Logger. This mode monitors the sampled data on the SIM-ME interface in order to:

- **Analyse the ATR** -- it determines the characteristics of the transmission protocol T=0.
- **Analyse the time intervals between the transitions** -- it determines the transmission rate (baudrate).
- **Convert physical transitions in bytes** -- it represents this information in a transition chart, which provides adequate graphical treatment of the electrical signals.
- **Convert streams of bytes into APDU** -- it represents the different command and response messages according to the parsing information contained in the *Definitions Tree* corresponding to the SIM card.

Data is displayed in a tree structure in the *INQSmart Log* window. INQSmart Logger can interpret data at two levels of complexity:

- **Level 1**-- At this level of complexity, the information is interpreted without considering either the syntax of the command/response pairs supported by the SIM card or other specific information about the SIM card operating system. In this case the analysis is limited to displaying the transitions chart of electrical signals and to provide the byte representation of the messages exchanged on the I/O line, without any further syntactical analysis.

- **Level 2**-- At this level of complexity, the INQSmart Logger knows at least the syntax of the command/response pairs supported by the SIM card, according to the parsing information given in the *Definitions Tree*. One can display the logged information according to both the INQSmart Log and the *Sequential Log* format. The latter displays the messages at the application level. The message data is split up in its composing data elements, e.g., CLA, INS, P1, P2, File Identifier. The message data is represented in the corresponding format, e.g., ASCII characters can be visualised in hexadecimal or text format.

## 5 INQSmart in Test Tool mode for SIM chip cards

The test environment for a SIM card using the INQSmart in Test Tool mode contains the following items (see Figure 8):

- **INQSmart software package configured in the Test Tool mode.**
- **PC running the INQSmart software package.**
- **An appropriated card reader in which the SIM card under test will be inserted either directly (ID-1 SIM) or using an adequate adapter (Plug-in SIM). The card reader is connected to the PC through a serial port, or a TCP/IP connection depending on the type of card reader.**
- **A SIM card, which is submitted to the software certification procedure.**

![INQSmart in Test Tool Mode](image1)

![SIM chip card under test](image2)

**Figure 8** INQSmart in Test Tool mode for SIM card testing

In this mode a command is firstly prepared in the IOBuffer of the *Card Tree* data structure, and sent to the SIM card under test. This event is recorded in the *Log*. The response received from the SIM card is parsed in the IOBuffer of the Card Tree data structure. The parsing is made based on the command
structure found in the Definitions Tree data structure. When a SIM Card Model data structure is included in the test environment, an emulation task is performed by this model producing the emulated response. The response received from the SIM card under test and the emulated responses are also recorded in the Log. A comparison is performed between the two responses and the result of testing is decided based on this comparison.

In this mode, several levels of test complexity are identified.

**Simple Test Tool**

In this mode, the INQSmart software provides help in generating manually SIM card commands using the Interactive Test Window. The responses received from the SIM card under test are parsed and thus made easy to read and understand, which is especially useful when BER-TLV formatting is used. Only simple tests are allowed since command parameters must be filled in manually and no image of the SIM card internal state exists. However this mode allows one to build very rapidly a simple ME simulation in order to test the interface with the SIM card, regarding the command/response syntax for example.

**Test Tool with Test Library**

In this mode, operations that are manually performed in the Simple Test Tool mode are replaced with complex testing procedures, which are delivered by INTEGRI in the GSM11.17 Test Library. In this way, complex processing for software certification can be performed, testing is faster, and tests may be replayed any time needed. These features are especially interesting for tests that are based on predefined command-response pairs.

**Test Tool with Card Model**

In this mode, the functions of a SIM card are written and compiled inside a DLL, which represents together with the Card Tree and the Definitions Tree the SIM Card Model. This is in fact the emulator of the SIM card. This allows complex and accurate testing of SIM card applications, providing a high degree of maintainability of the Test Library since:

- the responses of the SIM card under test can be compared with a reference produced by the emulator;
- the internal state of the SIM card is represented in full details in the Card Tree, allowing a very good intuition for the tester of the actual state of the SIM application under test;
- the internal state of the card can be used for computing complex command parameters involving cryptographical computations, e.g., message authentication codes (MAC), cryptograms, digital signatures, which can be used in the protocols describing the SIM application.

Once a SIM card emulator is written this can be included in the INQSmart running in the Emulator mode, without other development and thus at no extra cost. Therefore, the functionality of the emulator can be used twice in the testing project, if testing of both the SIM and the ME is performed by the same independent testing company:

- for SIM card testing, using the INQSmart in the Test Tool mode with SIM Card Model. The “GSM 11.17 certified” qualifier can be obtained if the tests in the GSM 11.17 Test Library are successfully passed.
- for ME testing, using the INQSmart in the Emulator mode. The “GSM 11.10-4 certified” qualifier can be obtained if the tests in the GSM 11.10-4 Test Library are successfully passed.

**Session components**

The following components will be found in a typical SIM certification session:

- **GSM 11.17 Test Library** -- This test library contains test scripts used to prepare and send commands to the SIM card under test.
- **ICC Interface** -- This object is used to store the parameters for the communication between the PC running the INQSmart software in the Test Tool mode and the card reader.
- **Definitions Tree** -- This data structure contains the syntax of the command/response pairs and the completion codes (the status words SW1, SW2) of the commands supported by the SIM card. This data structure represents the basic information for parsing the command/response messages exchanged between the SIM card under test and the INQSmart Test Tool. The Definitions Tree also
contains the message identification patterns of the commands accepted by the SIM card, as well as some card-specific values, tags and templates.

- **Card Tree** -- This data structure contains a template called `IOBuffer`, for outgoing commands and incoming or emulated responses. When INQSmart is used in the Test Tool mode with a SIM Card Model, the Card Tree also contains the file structure of the tested card. This structure is usually found under the `Master File` group and the `Transient` group. While the Master File group stores the permanent data of the model, the Transient group contains a set of temporary variables used during the processing carried out by the emulation functions of the SIM Card Model.

- **Log** -- This data structure records all relevant events during a testing session: outgoing commands, incoming and emulated responses, script execution status.

- **DLL** -- This DLL implements special parsing functions which are sometimes needed for some commands or responses. It may contain also emulation functions that, in the Test Tool mode with SIM Card Model, are used to compute emulated responses to be compared with the received responses from the SIM card under test. These emulation functions also update correspondingly the status of the emulated card in the Card Tree data structure.

### 6 INQSmart in Emulator Mode for ME

This section introduces the **Emulator mode** of the INQSmart software package. In this mode a SIM Card Model, shortly referred as **emulator**, is available. This emulator can be used to test the ME. In this case, the test operator does no longer need a large set of different physical SIM cards personalised differently in order to check the functionality of the ME under test in various circumstances. One can dynamically configure the structure of the files in the emulated SIM card, modify the content of the files, and change the behaviour of the emulated SIM card. The later feature allows the test operator to "overwrite" the normal response of the emulated SIM card in order to simulate error situations and to analyse the behaviour of the ME in each case. All the command/response transactions are represented and logged in a comprehensive way, which allows analysing in detail the transaction flow between the emulated SIM card and the real ME under test. Application designers can use the SIM card emulator to exercise how to program new SIM applications and to develop in an inexpensive way new protocols.

In order to set up a test environment using INQSmart in the Emulator mode, one needs the following items (see Figure 9):

- **INQSmart** software package configured in the Emulator mode.
- **PC** running the INQSmart software package.
- **Hardware adapter** (HWA) provided together with a probe connected to it through a multiple-wire cable. One side of the probe is going to be inserted into the ME under test. The contacts of this side of the probe can be seen as the contacts of the emulated SIM card. The other side of the probe, which is equipped with a chip card slot, is not used in this mode. The main role of the hardware adapter is to implement the transmission protocol T=0 between the emulated SIM card and the ME under test, which both comply with the ISO 7816 standard, regarding the physical and electrical characteristics. The hardware adapter is connected to the PC through a TCP/IP connection. The hardware adapter can also capture and log the whole sequence of command/response pairs exchanged between the emulated SIM card and the ME, in the same way as it was described in the section about the Logger mode of INQSmart.
- **The ME to be tested.**
- **In the Pass-Through mode**, a supplementary card reader and a real SIM card are needed since most of the emulation task is carried out by a real SIM card.

The command entering the SIM card emulator is first parsed in the `IOBuffer` register of the Card Tree, and this event is recorded in the Log. The parsing is generally made based only on the command structure found in the Definitions Tree, but some special parsing function written inside a DLL may be needed in some cases. A response template is then prepared in the `IOBuffer`, using information found in the Definitions Tree. The emulation task is then performed, and this depends on the mode of emulation that is chosen, as explained below. Once the response has been built, it is compiled and sent to the ME under test, and this is also recorded in the Log.
Emulator

In this mode, the emulated response is built by Event Scripts that are programmed in a simplified Test Library. Only simple emulation tasks are allowed since scripts are usually kept short. However, this mode allows one to build very rapidly a simple SIM card emulator in order to test the interface with the ME, regarding the command/response syntax for example.

Pass-Through Emulator

In this mode, a real SIM card is inserted in the probe of the hardware adapter, and commands issued by the ME under test are forwarded directly to this real SIM card. The response sent back by the real SIM card is then parsed in the IOBuffer register of the Card Tree and Event Scripts are run. The Event Scripts can be used to modify the response provided by the real SIM card, which proves to be very useful in ME testing, to study its behaviour in error conditions. Once modified, the response is compiled and sent to the ME. Introducing a delay in the communication is one of the possible applications, to test the ME’s robustness to long processing time.

Advantages introduced by this configuration are:

- a transaction log is obtained, similar to the one provided by an INQSmart Logger;
- the ME and the SIM card need not to be used at the same physical location since a TCP/IP link is used between the Hardware Adapter and the PC, and maybe also between the card reader and the PC (if the card reader supports TCP/IP connection). ME and SIM cards located at very distant places can thus be used together, which is helpful for example for testing a ME prototype against a SIM card prototype when they are developed by different providers.

Emulator with Card Model

In this mode, the SIM emulated functions are written and compiled inside a DLL. This allows complex and accurate emulation that are 100% similar to actual SIM cards, but for which no real SIM cards are needed, and that can thus be made available at an early stage in a project development. Also, prototyping of applications hosted by the SIM is possible at low cost. For running a software certification procedure of the ME according to GSM 11.10-4, INTEGRI can provide the GSM11.10-4 Test Library in the configuration of the INQSmart in the Emulator mode.
7 Conclusions

The software certification activity of SIM chip cards and ME is a complex task. The paper has proposed various test environments for SIM/ME software certification. The solution is based on hardware/software components in the INQSmart utility developed by INTEGRI. This testing tool is characterised by a high versatility, allowing for reducing the overall price for testing in a SIM/ME software development project. At the level of test management, the test tool provides for high maintainability of the test scripts, as well as the possibility to automate the execution of tests, especially important for regression testing.
Automated Testing Environment for SIM Application Toolkit

Cristian Radu

Dr. Cristian Radu is currently a Security Consultant with Integri in Belgium. Integri is an independent testing organisation active in the field of electronic payment systems and telecommunication networks. Prior to Integri, he was a researcher with the Computer Security and Industrial Cryptography group at the Catholic University of Leuven, Belgium. Since 1993 he has been directly involved in the design of cryptographic primitives and protocols for electronic banking. His interests are in computer security and industrial cryptography, electronic commerce, test design, and test automation for security software. He holds a PhD in Electrical Engineering from the Catholic University of Leuven, Belgium.

Email address: cradu@integri.be