The 4th Symposium on Architecture Oriented Formal Approaches to High Quality Software Development

The Application of VDM to the Industrial Development of Firmware for “FeliCa” Smart Card IC Chip

February 13, 2013

Taro KURITA

Sony Corporation and FeliCa Networks, Inc.
Agenda

- What is “Mobile FeliCa”
- Development of First Generation (“The Stone Age”)
- Development of Second Generation
  - Goal
  - Approach
  - Result
- Development of Third Generation
  - New Features and Style, Framework, etc.
- Summary and Current/Future Issues
• What is “Mobile FeliCa”

• Development of First Generation (“The Stone Age”)

• Development of Second Generation
  – Goal
  – Approach
  – Result

• Development of Third Generation
  – New Features and Style, Framework, etc.

• Summary and Current/Future Issues
What is “Mobile FeliCa”

- “FeliCa” is a contactless IC card technology widely used in Japan.
- FeliCa is developed and promoted by Sony Corporation.
- FeliCa uses Near Field Communication (NFC) technology.
- FeliCa is used for electric money, train tickets, identification, door keys and so on.
- Today, “Mobile FeliCa” IC chips were embedded in over 125 million mobile phones.
The mobile FeliCa system is comprised of
- mobile phones with a FeliCa IC chip,
- FeliCa servers connected to the mobile telecom network,
- and FeliCa reader/writers.
The mobile FeliCa system is comprised of
- mobile phones with a FeliCa IC chip,
- FeliCa servers connected to the mobile telecom network,
- and FeliCa reader/writers.

- The FeliCa file system specification that defines the basic data structure
- Wireless and wired interfaces of the Mobile FeliCa smart card
- About 100 commands which are the basis of the FeliCa technology
- Security specifications combined with file system and command specifications
The mobile FeliCa system is comprised of
- mobile phones with a FeliCa IC chip,
- FeliCa servers connected to the mobile telecom network,
- and FeliCa reader/writers.

The Mobile FeliCa IC chip firmware contains the secure file system and the communications protocol.

It has firewall functions that enable multiple services in the Mobile FeliCa IC chip, such as electric money, train tickets and identification.
• What is “Mobile FeliCa”

• Development of First Generation (“The Stone Age”)

• Development of Second Generation
  – Goal
  – Approach
  – Result

• Development of Third Generation
  – New Features and Style, Framework, etc.

• Summary and Current/Future Issues
Development of First Generation ("The Stone Age")

- 2001 - 2004
- Death march project with vague specifications
- Then, we decided to study formal methods and describe rigorous specifications.
• What is “Mobile FeliCa”

• Development of First Generation (“The Stone Age”)

• Development of Second Generation
  – Goal
  – Approach
  – Result

• Development of Third Generation
  – New Features and Style, Framework, etc.

• Summary and Current/Future Issues
Development of Second Generation

- 2004 - 2006 (First Release)
- 2006 - 2010 (Modified Release)

- We have developed the “Mobile FeliCa” IC Chip firmware specifications with a formal method using VDM++ and VDMTools.

- We use Level 0 Formal Methods.

- We described and tested formal specifications without proof.

- Thanks to the formal methods, there are no problems related to the software specifications since the first release in 2006.
Goals for Applying the VDM

• Creating precise specifications
• Enhancing the quality of deliverables at the design phase
• Improving development processes
• Testing completely with different approaches
• Activating communication between engineers
Project Duration and Members

- The project duration was **three years and three months**. It finished **on schedule**.

- There were **50-60 members**. The average age was about **30 years old**.

- There were **no members who had knowledge of or experience with formal methods**.
Teams for Overall Development

We organized three teams:

- **Specification** Team of 5-20 members
- **Firmware Implementation** Team of 15-20 members
- **Testing** Team of 25-35 members
Artifacts for Overall Development

- Developed products are shown with solid lines and tools used in development are shown with dashed lines.
- We described and tested specifications of external behavior using VDM++.
- We developed executable formal specifications.
Non-functional specifications such as performance and reliability were written in the natural language separately from the formal specifications.

This is the framework for describing and testing specifications that are based on the basic data structure, interfaces and templates.
Process for Overall Development

1. Writing of High Level Specifications (Natural Language, UML)
2. Description of Formal Specifications
3. Unit Test
   - NG
   - OK
4. F/W Design, Implementation
   - NG
   - OK
5. F/W Unit Test
   - NG
   - OK
6. Test Case Design
   - NG
   - OK
7. Specification Test, “Test Cases” Test
   - NG
   - OK
8. F/W Test
   - NG
   - OK

Process: OK → NG → OK → NG → OK → NG → OK
Results

- 383 pages of a protocol manual written in the natural language
- 677 pages of an external specification document written in the formal specification language

- Our formal specifications are about 100,000 LOC including test cases (about 60,000 LOC) and comments.
- Using this specifications, we implemented the C++ code of about 110,000 LOC, inclusive of comments.
The average productivity of VDM++ code for the formal specifications was about 1,900 LOC per engineer per month.
Formal Specification Errors in Specification Phase

<table>
<thead>
<tr>
<th>Phase of Development Process</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Describing Specifications</td>
<td>162</td>
</tr>
<tr>
<td>Executing and Unit Testing Specifications</td>
<td>116</td>
</tr>
<tr>
<td>Reviewing Specifications</td>
<td>93</td>
</tr>
<tr>
<td>Communicating with Firmware Engineers</td>
<td>69</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>440</strong></td>
</tr>
</tbody>
</table>

Debug Density = 440/40,000 = about 11 errors/kLOC

The formal method contributes to enhancing the quality of deliverables at the early stage of the development process.
## Errors in Firmware Implementation Phase

<table>
<thead>
<tr>
<th>Reason for Errors</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Missing description</td>
<td>0.2%</td>
</tr>
<tr>
<td>Erroneous description</td>
<td>0%</td>
</tr>
<tr>
<td>Unclear description</td>
<td>1.8%</td>
</tr>
<tr>
<td>Oversight</td>
<td>5.6%</td>
</tr>
<tr>
<td>Insufficient understanding</td>
<td>10.7%</td>
</tr>
<tr>
<td>Insufficient confirmation</td>
<td>0%</td>
</tr>
<tr>
<td>Failure of change propagation</td>
<td>0.2%</td>
</tr>
<tr>
<td>Others (reasons unrelated to specifications)</td>
<td>81.5%</td>
</tr>
</tbody>
</table>
It can be said that we have successfully described the specifications in a precise way.

The formal methods are useful for finding errors in the early stages of development.
Errors in Firmware Implementation Phase

- On the other hand, the total percentage of “oversight” errors and “insufficient understanding” errors was 16.3%.

- This was due to the fact that the separations between the actual specifications and the code required to execute the specifications was unclear.

Future Issue: Easy-to-read
Test Scheme

- **Formal Specifications Test**
  - Distribution
  - Unit Test
  - (1) Formal Specifications Test
  - (2) "Test Scripts" Test

- **Firmware Engineers**
  - F/W Design
  - F/W 1
  - F/W n
  - Unit Test

- **Test Engineers**
  - Test Cases
  - Black Box Test Scripts
  - IC Card Simulator, "Random Test" Tool
  - Unit Test

- **IC Chip Test**
  - IC Chip 1
  - IC Chip n
  - Unit Test

- **Specifiers**
  - Formal Specifications
  - Unit Test
Test Results

[Unit Testing of Formal Specifications]
- The line coverage rate of the VDM++ formal specifications by unit testing was 82%.

[Black-box Testing of Formal Specifications and Implementation]
- The line coverage rate of the formal specifications by black-box testing and visual inspection was 100%. (This is checking of test cases of black-box testing.)

[“Random Testing” of Implementation]
- “Random Testing” is a continuous testing.
- The test tool sends random commands continuously to the test target and checks whether the test target sends back correct responses.
- Random test tool (designed using Ruby language) is transcribed from VDM++ model by hand.

- By carrying out about 7,000 black-box tests and over 100 million random tests, the high quality of IC chips was achieved.
Conclusions from Second Generation

- The application of the level 0 formal method was highly effective for the successful completion of our project on schedule.

- The formal method contributes to the quality of deliverables at the early stage of the development process.

- And, the formal method appears to have activated and encouraged communication between engineers, which is vital for software development.

- It is necessary to pay attention to not only executable features, but also the readability of specifications.

- Specifications which are referred to by all project members need to be simple, so that it can be read without stress.
• What is “Mobile FeliCa”

• Development of First Generation (“The Stone Age”)

• Development of Second Generation
  – Goal
  – Approach
  – Result

• Development of Third Generation
  – New Features and Style, Framework, etc.

• Summary and Current/Future Issues
Development of Third Generation

2007 - 2013

New Features:

- Update for Advanced Encryption Standard (AES)
- Adapt to Near Field Communication (NFC) to use Mobile Wallet worldwide!
Style of Specification

2nd Generation:

We cannot separate FeliCa easy-to-read specifications and description of executable specifications.

We only used ASCII code characters in VDM models.

3rd Generation:

We are describing FeliCa specifications by data structure class and pre/post conditions and invariants.

We wrote specifications using Kanji (Unicode) characters.
International Common Criteria Security Level

ISO/IEC 15408 Common Criteria:

   The world's standard for evaluating security products and systems.

2nd Generation:

   Evaluation Assurance Level 4+ that is methodically designed, tested and reviewed

3rd Generation:

   Evaluation Assurance Level 5+ that is semi-formally designed and tested
The Evaluation Assurance Levels (EAL1 through EAL7) of an IT product or system is a grade assigned following the completion of a common criteria security evaluation:

EAL1: Functionally Tested
EAL2: Structurally Tested
EAL3: Methodically Tested and Checked
EAL4: Methodically Designed, Tested and Reviewed
EAL5: Semi-formally Designed and Tested
EAL6: Semi-formally Verified Design and Tested
EAL7: Formally Verified Design and Tested
• What is “Mobile FeliCa”

• Development of First Generation (“The Stone Age”)

• Development of Second Generation
  – Goal
  – Approach
  – Result

• Development of Third Generation
  – New Features and Style, Framework, etc.

• Summary and Current/Future Issues
Current/Feature Issues

- Negotiating with stakeholders who do not use formal methods.
- Translating and testing user's manuals that is based on formal specifications.
- Defining effective combinations of formal and informal specifications.
- Description of formal specifications suitable for embedded systems and code generation.
- Framework for describing specifications that are easy-to-read and executable.
- Specifications that firmware engineers and test engineers feel familiar with and comfortable reading.
We conclude that formal methods are suitable for reaching high quality and fit within the broader Japanese traditional philosophy of “Kaizen” – continuously improving the process step by step, working closely together with team members.

- Gradual, orderly, and continuous improvement!
- Ongoing improvement involving everyone!