MULTOS Development Tools

To develop or not to develop?

…that is the question that a multitude of people ask me at every show. Should I develop my application for MULTOS? How difficult is it going to be? What is available?

Today there are numerous smartcard applications available from a multitude of different vendors in all kinds of application areas but the majority of them are only available on a subset of proprietary cards. With the advent of open, standard, multi application cards these application developers are worried that they may be missing out.

- Are their current customers going to start asking for their products on these open, multi-application cards?
- Which cards are they going to want? JavaCard? MULTOS? SCfW?
- Which platform should they develop for?

My answer to this is always the same… develop for as many platforms as you can. If you can offer your application on all current standard platforms you have a marketing advantage over your competitors.

Help! Where do I start?

Out of all three multi application operating systems, with which should I start? Lets start by looking at what languages are available…

My application is coded in Assembler

MULTOS is the only platform that has an easy to use assembler language. MULTOS smart card applications were originally developed in MEL (MULTOS Executable Language), which contains typical assembler instructions plus “smart card friendly functions” called *primitives.*

My application is coded in C

MULTOS is the only platform that currently has a C compiler. This is the most common embedded language at the moment. The SwiftC development tool from SwiftCard is an ANSI compliant compiler that allows you to very quickly port an existing application to the MULTOS operating system. Proof of concept of most ports can be completed within two weeks. Scott Guthery of Mobile-Mind recently ported a DOS FAT file system from GNU source code in two days having never used Swiftcard before. Of the Swiftcard C compiler Scott said "The compiler is quite rigorous and caught lots of ambiguities. The generated MEL code is very efficient. In size I found it to be only about 25% larger than 8051 native code generated by the best 8051 C compiler on the market."

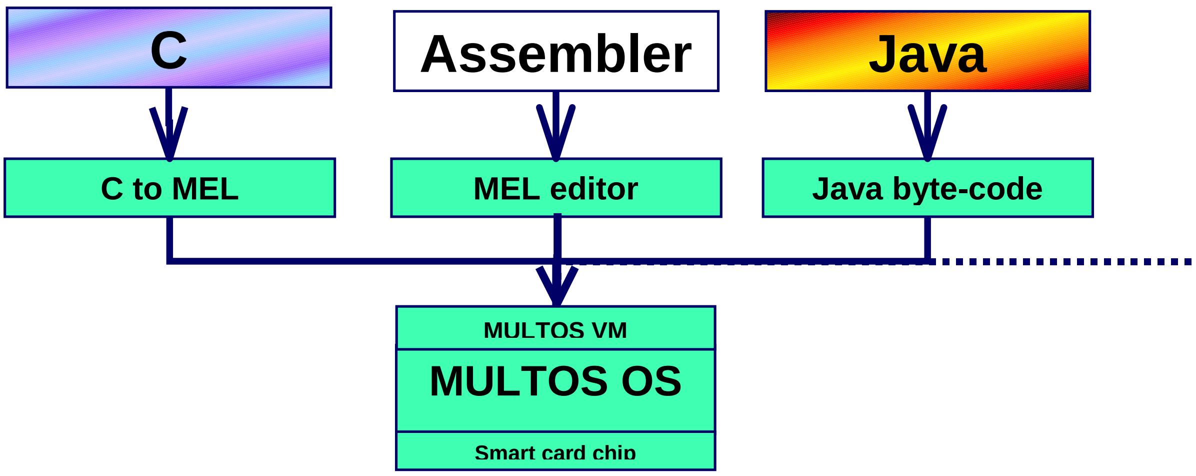
My application is coded in Java

Both JavaCard and MULTOS can support the Java language. In both cases a Java compiler translates the source to Java class. For JavaCard the classes are converted to JavaCard byte code. For MULTOS, the SwiftJ compiler from SwiftCard translates the Java classes (or Basic, or Modula2) to MEL code. It is worth noting that code written in Java for one target operating system (e.g. JavaCard) will **not** work on the other (e.g. MULTOS) without some modification.

My application is coded in Visual Basic

Really! Smartcard for Windows has chosen this as their application development language of choice. To make MULTOS accessible to the VB community SwiftCard Technology are currently working on a Visual Basic to MEL translator.

What we can see here is that using MULTOS we are not limited to using a single language for all applications, unlike other systems.



Language is not the issue

Perhaps the choice of which operating system to start with is not dependent on the language, maybe because the code is going to be rewritten anyway. Perhaps the ease of writing the application is the key factor. This is going to be based around two key things… the simplicity of writing for the card, and how straightforward the development tools are to use.

In the next section we will see what tools are available for MULTOS and focus on one in particular.

Simplicity of writing for the card

When writing MULTOS applications for the first time, most people who have travelled this path have found the MULTOS route a lot smoother than the others.

For example: MULTOS handles all of the communications between the card and the terminal after the developer has called one primitive declaring the ISO case. The operating system will even take care of the protocol being used. Once an application developer has declared that the command is ISO Case 4 with the call CheckCase(4); MULTOS will do what ever is necessary to get the information from the terminal. When the application exits MULTOS will pass back any information stored in the PUBLIC memory area to the terminal. The developer is therefore left to concentrate on coding the application and not fighting the operating system.

MULTOS Development Tools

Let's have a look at what development tools are available for MULTOS.

In the beginning…

there was *MDS v1 (and its derivatives)*

boldly going where no tool had gone before.

MDS had a good MEL assembler, but the baseline was MULTOS version 3 and so it could not be used to develop applications that make used of the added functionality in MULTOS version 4. Two bolt on modules were also available, a C compiler and a loading and deleting module. Both of these options functioned but were less than satisfactory for serious smartcard development. These versions of MDS were only compatible with Windows 95/98 forcing some developers to support a second PC.

Because of the initial lack of an Application Loader, MAOSCO developed AppLoader - a freeware piece of code. AppLoader allowed applications to be loaded to and deleted from a MULTOS card - including the creating and loading of a directory record so that the contents of the card could be viewed.

Now…

There are three tools

- MDS from GTI-UK v2.0
  www.gti-uk.com
- MULTOS Development Tools from Hitachi Micro Systems Europe v1.2
  www.hitachi-eu.com/smartcard
- SwiftCard/MEL from SwiftCard Technology v1.2
  www.swiftcardtech.com

All tools are now baseline MULTOS 4, therefore supporting both MULTOS version 3 and MULTOS version 4 application development.

All tools:

- contain a C compiler, assembler, debugger, card simulator and (PC/SC compliant) terminal simulator
- have application load/delete capabilities.
- are Windows 95/98/NT/2000 compatible.

MDS v2.00

MDS v2.00 has been improved in many ways based on a lot of feedback from many different companies previously using v1.24 of the tools. The C compiler has been optimised to give better MEL code size and thus allow faster application development. Although no C source debugging is available in v2.00 it will be in later versions. The assembler speed has been improved by 200 times and the load and delete mechanism, as well as being improved, now supports MULTOS version 4 cards. The SoftTerminal terminal emulator can be used to send smartcard commands to the card or directly to the debugger and can also simulate card to card transactions using a scripting capability implemented in interpreted C source code.

Hitachi Development Tools v1.2

The Hitachi tools use a slightly different Assembler syntax to the other two, called MAL. This is very similar to the MDS syntax and converting from one tool to another is not too difficult. Hitachi currently have a beta version of the C compiler which will be released in August 2000 with C level source debugging in the IDE. One of the benefits of Hitachi is their complete solution. Hitachi can support a company dealing in MULTOS in many ways: They supply live and development cards, card readers, terminals, development tools as well as integrated and support systems.

SwiftCard/MEL v1.2

This tool has seen the most action in recent months with the addition of a Java compiler (SwiftJ) to allow simple conversion of applications originally written for

JavaCard. This compiler also supports other languages such asModula-2. SwiftCard Technology (SCT) is also currently working on a Visual Basic translator.

The SwiftCard tools are MDS and Hitachi MALT mnemonics and syntax compatible which means that existing applications written in MDS can be compiled in SwiftCard.

The assembler also contains extended features, giving MEL more power. The C compiler is highly optimising both on local and global code giving impressive final code sizes of about 30% bigger that the same applications written in native code for the underlying chip, or written in MEL code.

The debugger and simulator features all of the MULTOS functions including delegation (calling one application on the card from another), transaction protection (automatic data protection over multiple writes) and cryptography.

The simulator supports source-level debugging of C, Java and MEL and the tools include a linker and archiver allowing Java, C, MEL etc. modules to be combined into one application.

The tool include a powerful Load/Delete mechanism (under T=0 and T=1 protocols) which can use pre-selected load/delete certificates or can even generate the certificate, specific to the application, on the fly.

As well as a Terminal Simulator there is a scripting mechanism which allows comprehensive test scripts to be written in VC++/Borland C/Delphi/VB (examples included)

In v1.3 a development IDE will be included which has syntax highlighting.

How do I get started?

Having easy-to-use development tools and multi-application operating system cards is all well and good, but how difficult is it for someone to start developing applications?

The first thing that a prospective developer must do is register using the online form at www.multostechnet.com/dla/

Registration is simple and *free* and gives the user access to all of the MULTOS development documentation, code samples, discussion forum, freeware and free technical support.

Before any development can really commence various things much be purchased:

- MULTOS development cards (£20 each)
- Development Tool (~£3000) (7 day evaluations available from company websites)
- AppLoader (Free) - not strictly necessary but useful
- Card Reader (~£50) (although most PC/SC readers work with the tools and AppLoader)

From this point on the MULTOS and tool documentation should allow for rapid application development. Technical Support is available from MAOSCO via both phone and email, including consultancy services.

Conclusion

The ability to use high level languages such as C, Java and Visual Basic when writing MULTOS applications allows developers, new to smart cards, to rapidly implement new applications in the language of their choice. Thanks to this, there are already more than 50 MULTOS applications available off the shelf, with many more under development.

Published by

MAOSCO Limited,
47-53 Cannon Street,
London, EC4M 5SQ,
United Kingdom.

General Enquiries

Email: Customer.Services@Multos.com
Tel: +44 (0) 20 7557 5565
Fax: +44 (0) 20 7557 5575
Web: http://www.multos.com

Technical Enquiries

Email: Dev.Support@Multos.com
Tel: +44 (0) 20 7557 5560
Fax: +44 (0) 20 7557 5570
Web: http://www.multostechnet.com

Document References

All references to other available documentation is followed by the document acronym in square [ ] brackets. Details of the content of these documents can be found in the MULTOS Guide to Documentation, and the latest versions are always available from the MULTOS TechNet web site (http://www.multostechnet.com).

Data References

All references to MULTOS data can be cross-referenced to the MULTOS Data Dictionary available in the back of the MULTOS Developers Reference Manual.