Mobile Application Development for Multi-platform Deployment

Ms. Pranita G. Malve
Department of Computer Science & Engineering
Prof. Ram Meghe Institute of Technology and Research, Amravati, India
Email: pranitamalve@gmail.com
Guide: Prof. S. R. Gupta

ABSTRACT

One of the biggest challenges to application development for mobile platforms is due to device fragmentation. Porting a single application on different mobile platforms multiplies expenses and is time taking. We aim to develop a framework to automatically transform an application written in a particular mobile platform into an application that can be ported to other mobile platforms. We intend to develop a graphical modeling language which is specific to mobile platforms and come up with algorithms to generate the application source code from the graphical model.

1.0 PROBLEM STATEMENT

Today's current mobile device marketplace contains many different software platforms such as Windows Mobile, Symbian, Palm, Java Mobile, IPhone and Android. Many of the applications today are only available on the platform they are developed for. Trying to develop the same application across all platforms creates expensive problems when individually porting the code to each platform. Most of the development time for a mobile project is spent working on code porting and deployment instead of the initial development. So, we are researching on developing a framework that would allow applications to be written once and deployed to multiple mobile platforms.

1.1 Business Motivation

There are three main reasons behind this developing this framework to achieve cross-platform portability of the mobile applications. Firstly, the new approach will cut down on development time. It will reduce the amount of time spent porting the application as well as debugging the application on each platform which will make the development more efficient. Secondly, new software will be available on different devices and thus increasing its availability and profitability. Thirdly, Developers will be able to focus on content, functionality and features instead of fixing issues with the porting and deployment of the application.

1.2 Research Problem

We need to separate the business logic from the actual implementation. We are researching on developing a graphical modeling language with which we can model the business logic and application requirements independent of any constraints imposed by a particular mobile platform. After-wards we need to develop platform specific algorithms for the conversion of graphical model into code.

2.0 LITERATURE REVIEW

While developing the cross-platform portability for mobile applications, the major concern is to make the mobile applications work in the heterogeneous environments. Some studies to be considered in this research are Object Migration in heterogeneous environments, different implementations of Middleware to migrate with an adaptation of its state and implementation at runtime.

Systems, which rely on language-dependent serialization formats, do not support heterogeneous environments. But *Peter and Guyennet* [4] suggest some approaches to achieve object mobility using CORBA in large-scale systems. Even good amount of work has been done on Web service object migration by *Hammerschmidt and Linnemann* [5]. However, this migration in the system is based on Java serialization, which does not support dynamic adaptation of state and implementation code.

As *Hallsteinsen et. al.* [6] suggested a framework MADAM, a work on flexible planning-based middleware for context-aware mobile application. This component framework has applications which are composed of components. These components can be reconfigured as per their context. Their adaptation is restricted to a custom component framework.

Kunze et. al. [2] suggested DEMAC which focuses on distributed execution of mobile processes using process description language. This is interpreted by infrastructure components for process execution. But, DEMAC does not support self-adaptive migration as only process description is transferred. Since it does not support an MDA development approach, parsing mobile process at each step leads to higher processing resources.

Ishikawa et al. [3] describe the mobile processes with BPEL. They describe a proprietary behavior description for mobile agent systems, which can migrate Web services on other machines. However, context is not supported sufficiently.

Schmidt and Hauck [1] present a design of SAMProc, a middleware for self-adaptive mobile processes in heterogeneous environments. This supports complete migration of services and allows adaptation of the application to the current device's application context. The self-adaptive mobile processes allow the abstract specification of an application's functionality as well as interactions and deployment aspects.

3.0 SOLUTION APPROACHES

The solution approach to accomplish the cross-platform portability of mobile application would include application development using a Model-driven Architecture approach. This would include determining the software incompatibility set between different platforms and then developing Plug-ins specific for each platform so as to meet the software incompatibilities. So, the applications would be developed using the common set of software functionality existing between different mobile platforms and then wiring these modules with platform specific Plug-ins to produce the full-fledged application.

The developer has just to code the application logic, interaction and deployment aspects, and functionality in each of the component.

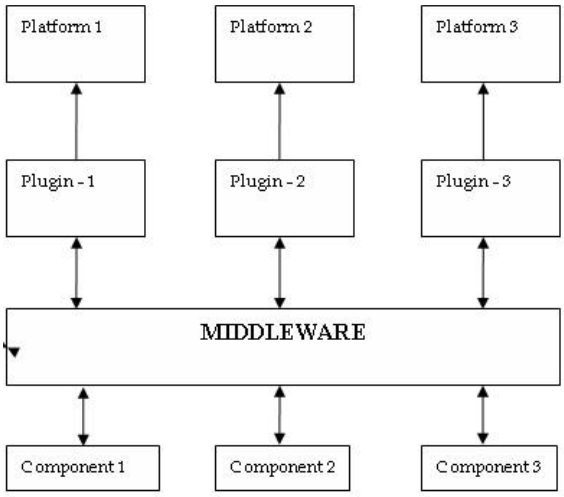

Figure I: Code generation process for cross-platform development.

These components would specifically be coded using a platform-independent set of libraries. Then, the mobile processes are converted into adaptive processes by the Middleware part. This part may include some kind of MDA language or a scripting language like BPEL to convert the processes into self-adaptive objects that are able to migrate over the platforms. These applications with additions of link libraries from the plug-ins would automatically generate a platform-specific mobile application.

3.1 MDA Approach

MDA provides an open, vendor-neutral approach to the challenges of business and technology change. MDA separates business and application logic from underlying platform technology. Platform independent models specify the business functionality and behavior of an application separate from a technology specific code that implements it.

Platform Independent Model: Based on common software support we specify various aspects of our application using a suitable modeling language rather than programming manually. These models would later on be translated into executable code for a specific platform. This transformation is implemented using a Model transformation which is an iterative process.

Platform Specific Model which specifies how the system is implemented. It determines how the PIM executes in the target deployment environment. A PIM is used as foundation for mapping into one or more platform specific models. Such a PSM describes in detail how the PIM is implemented on a specific platform, or in a certain technology. PSMs are also expressed in UML adding constraints and implementation details.

Platform Model is the final product and corresponding to code written in a specific programming language for a specific application.

3.2 Multi-language, multi-target compiler

We propose to build a multi-language, multi-target source to source compiler. This compiler could take the source code of one language/platform as input and generate executable code for other platforms. This compiler would have a front end for each target platform. This front end would perform the syntactic and semantic analysis and translation to a lower level representation of the source code. The output of the front end would be a parse tree or Abstract Syntax Tree. The middle-end of this compiler

would perform code optimization, dead code elimination, dynamic analysis and other code fix-ups.

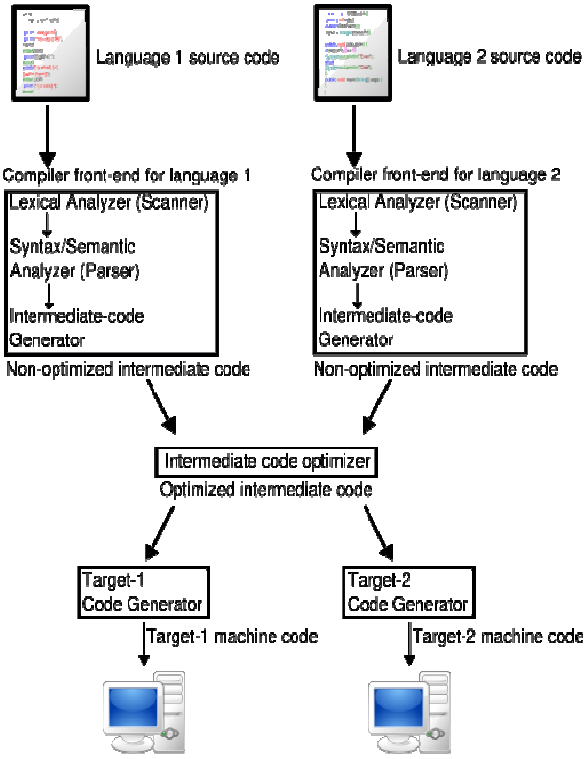

Figure II: Operation of a typical multi-language, multi-target compiler.

This optimized code will form the input of the compiler backend. This backend would be different from the normal compiler backend in the sense that it would not generate assembly code but rather would do code transformation so as to generate code for different platforms. The generated code could then be compiled against the platform specific compilers to produce the executable.

3.3 Graphical Modeling Tool

In order to apply and test the design concepts, a graphical modeling tool called Mobile Application Modeler (MobiAppModeler) will be created. Following are the basic steps that would be done in order to accomplish this.

1. The MobiAppModeler tool would be created such that users could specify the services and functionality that they want in their application.

Figure III: Graphical Modelling Tool

2. This tool will have a set of GUI building blocks such as screens, buttons, textbox, combobox, frames, panels etc.
3. Using these GUI elements the user would be able to build each of the screen, its contents, set the properties of GUI elements used, set events and triggers and set the transition between different screens in their application.
4. Internally a state machine will be built by the tool which will record all the events and transitions.
5. Finally, the tool would give the user option to generate code for any of the following mobile platforms: Android, JavaME, iPhone
6. The generated application source code will be compiled and generate an executable that could be run on the selected mobile platform.

The code would be generated based on the state machine developed for the particular application. Each state in the state machine would represent a GUI element and the transition from a given state to another would be dependent on some decision. The code to create GUI elements and handle their events would be different for each platform.

4.0 TASKS IDENTIFIED

The framework described above has to be developed in different modules. But for now, the first and foremost thing is to identify the functions and device configurations that are needed to implement a basic functionality using different platforms in the mobile phone.

The objective would be to identify the functions and the device configurations that are common between the different platforms. These functions would contribute to the platform-independent libraries that the developer would use to code his application logic and functionality.

I will start to build the Graphical Modeler tool incrementally in several iterations of development.

5.0 CONCLUSION

A viable solution has been identified to achieve cross-platform portability of mobile application and Graphical modeling tool that converts the states to comprehensive code solves the portability issue to a larger extent. However, portability on increasing code complexity stays open for further research.

REFERENCES

[1] Holger Schmidt and Franz J. Hauck. 2007. SAMProc: middleware for self-adaptive mobile processes in heterogeneous ubiquitous environments. In Proceedings of the 4th on Middleware doctoral symposium (MDS '07). ACM, New York, NY, USA.

[2] Christian P. Kunze, Sonja Zaplata, and Winfried Lamersdorf. 2006. Mobile process description and execution. In Proceedings of the 6th IFIP WG 6.1 international conference on Distributed Applications and Interoperable Systems (DAIS'06), Frank Eliassen and Alberto Montresor (Eds.). Springer-Verlag, Berlin, Heidelberg, 32-47.

[3]Fuyuki Ishikawa, Nobukazu Yoshioka, and Shinichi Honiden. 2005. Mobile agent system for Web service integration in pervasive network. *Syst. Comput. Japan* 36, 11 (October 2005).

[4] Yvan Peter and Herv Guyennet. 2000. Object mobility in large scale systems. *Cluster Computing* 3, 2 (April 2000), 75-82.

[5] Beda Christoph Hammerschmidt and Volker Linnemann. Migratable Web Services: Increasing Performance and Privacy in Service Oriented Architectures. In IADIS Int. J. on Comp. Science. and Info. Sys., 2006.

[6] Mourad Alia, Frank Eliassen, Svein Hallsteinsen, and Erlend Stav. 2006. MADAM: towards a flexible planning-based middleware. In *Proceedings of the 2006 international workshop on Self-adaptation and self-managing systems* (SEAMS '06). ACM, New York, NY, USA, 96-96.