Towards the Integration of UPES and UML-RT for Developing an Embedded Software

Chung-Hao Wu Yong-Yi FanJiang
Department of Computer Science and Information Engineering
Fu Jen Catholic University, Taiwan ROC
haoka96@csie.fju.edu.tw *yyfanj@csie.fju.edu.tw*

Abstract *In the Embedded system area, there are many development processes have a completely combination to the UML-RT. The UPES (Unified Process for Embedded System) is a development process which based on platform-based design principles and UML 2.0. In this paper, we propose a workflow which utilizes the UML-RT and the UPES process for embedded software development. The capsule collaboration diagram and statechart are used to strengthen platform model in the UPES process. We also implement an example of digital photo frame to demonstrate our proposed workflow.*

Keywords: UPES, UML-RT, platform-based design, embedded software.

1. Introduction

Many products, such as the smart phone, DVD player, cars or peripherals of internet, contain software today [13]. In order to make the convenience of human's life, more products which containing embedded software to be developed. However, cause of people's needs, the complexity of the embedded software is growth up simultaneously. The growing dominance of software development in embedded systems design requires a careful look at the latest methods for software analysis and design. For these reasons, to reduce costs and marketing on time will become an important issue in embedded software development process.

There are many embedded software development processes and modeling languages indentified. Therefore, how to development an embedded system systematic still confuse the system designers. Platform-based design [2] is constantly used in the industrial field. The paper of E. Riccobene *et al*. [1] present a develop process of UPES (Unified Process for ES) which based on the Platform- based design. The process modeled by Y-development chart uses the classical UP (Unified Process) to define the *Application model* combined with *Platform model*, and maps the two models n the component diagram or deployment diagram.

In the modeling languages, UML is the most favorite one which used in the area of software illustration. The Unified Modeling Language (UML) [3] is a general-purpose modeling language for specifying, visualizing, constructing and documenting the artifacts of software system. UML has been used extensively in various fields. However, UML is lack of specialist concepts for particular application domains. UML-RT [4] is an UML extension for real time and embedded system modeling. The UML-RT notation emphasizes a hierarchical system organization. We focus on the description of structure by capsule collaboration diagram and description of behavior by statechart.

In this paper, we adopt the UML-RT notation integrated into UPES process and base on the proposed process to model and implement a digital photo frame for demonstration.

The remainder of this paper is organized as follows. In Section 2, we list some related work. The design process UPES and UML-RT are briefly described in section 3 and section 4, respectively. A design workflow combines the UPES and UML-RT is depicted in section 5. A digital photo frame (DPF) is provided as an example to illustrate our approach in section 6, and the conclusion and future plan are given in section 7.

2. Related Work

The reference documents of UPES and UML-RT is referring from [9][10][11]. A number of researches adopt the UML-RT to develop the embedded system [4][12]. In the paper of Qiang Zhu [12], they use the concept of platform-based design and the Y-development chart to model the development process SLOOP (System Level design with Object-Oriented Process). SLOOP is according with OOAD development techniques to avoid the risk of functionality, and use according the extension notation UML to help analyzing and designing. The SLOOP contains four models: Conceptual Model for describe the requirements, Functional Model focus on the structure without consider the physical architecture, Architectural Model present the architecture and physical resource, and Performance Model maps the result of Functional Model and Architectural Model.

The UML-RT extensions are used to construct the hardware structure of Platform model. Which the UML-RT is combines the UML1.1 modeling concept and special modeling constructs and formalisms originally implemented in ObjecTime developer and defined in the ROOM [10] language. The ROOM is a visual modeling language with formal semantics, developed by the ObjecTime. It is optimized for visualizing and documenting the real-time system. It also supports automatic code generation to ensure reliability and increase productivity. The ROOM actor structure diagram is used to explicitly represent structural design patterns. The ROOMchart is a special hierarchical type state machine which is used to capture reactive behavior of actors.

HASoC (Hardware and Software Objects on Chip) [3][4] methodology is based-on RUP process. HASoC starts with an uncommitted model and committed model to partitioning the abstract executable model into hardware and software implementations, and then mapped onto system platform.

In the [2], using the well-known pattern of a information system architecture to develop the embedded system. The system is separated into three layers of user interface, control logic, and process interface (database). By utilizing the test interfaces and simulators, the variant design is verified.

S.-U. Jeon *et al.* [5] provide a new viewpoint to model the behavior of embedded software. In the Behavior Modeling phase, they describe the behavior with interaction overview diagram and sequence diagrams. By using Interaction-base modeling, the global interaction of embedded system is visible.

3. UPES (Unified Process for ES)

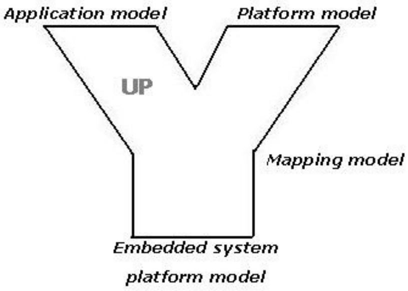

Figure 1: The UPES Y-chart

UPES is a design process which base on platform-based model to construct an embedded system form natural language description to the physical system. UPES is a MDA-style classical Y-development cycle (see figure 1). It consists of two sides - the Application model and the Platform model. The Application model made of the conventional UP [1] process for software development, to define the first executable model. The Platform model is a process which selects from several hardware components to an available hardware platform then modeled it with an appropriate modeling language (i.e. the UML extensions for real-time system etc).

When the software service and the hardware platform to be decided, we will map a reference model in the intersection. The Mapping model is described the mapping model of the SW and HW components, which describe with UML component and deployment diagrams.

The meet-in-the-middle process [11][13] is an iterative activity which to configure a platform model according to given HW/SW partitioning, then check wherever the QoS of this system is satisfied. More details of Platform-based design and UPES (see [9][11][13]).

4. UML-RT

UML is a modeling language for object-oriented software. In its basic form, it is useful to specify the general-purpose software systems, but lacks specialist concepts for particular application domains. Hence, a number of specialist extensions have been proposed. In the real-time system, we considered that the UML-RT is applicable to describe the hardware.

The most important UML-RT notation is the capsules, which are complex, concurrent, and possibly distributed active architectural components. In the behavior view of a capsule, the internal function can be represented by a statechart. A port is a physical part of the implementation of a capsule that mediates the interaction of the capsule with others capsule. Ports realize protocols, which define the valid flow of signals between connected ports of capsules. Finally, the connectors capture the key communication relationships between capsules.

In the description of UML-RT notations, the execution model constructed of the port which is transmit the signal and the statechart of single capsule. The sub-capsules are combined with its statechart by connectors in a complex capsule. The state machine, the sub-capsule, and their connections network represent parts of the implementation of the capsule.

5. The design flow of UPES combined with UML-RT

Figure 2 summarizes the phases of our design flow which contains three models: *Application model*, *Platform model*, and *Embedded System Platform model*. In Application model, first we follow the UP software development process to define the use case and the external event for our application, and then use the use case diagram and sequence diagram to describe how to transmit signals between actors and system. In the Platform model, we use the capsule collaboration diagram and statechart to modeling the hardware components and the behavior in the system. In the meet-in-the-middle phase, we adopt the Qvfb (Qt visual frame buffer) [8] mechanism to develop and test. Finally, the deployment diagram is used to present the Application model to the Platform model.

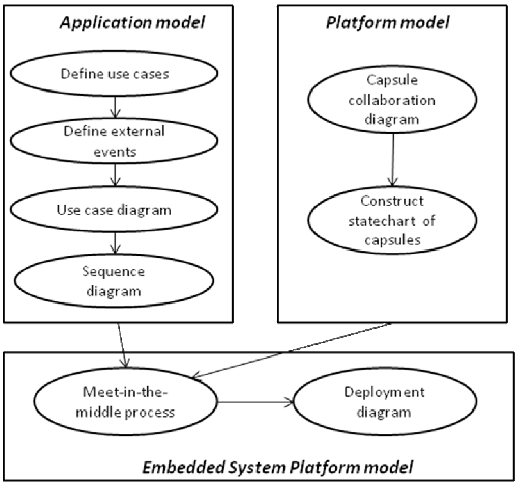

Figure 2: The overview of our approach

6. Case study: digital photo frame

In the section, we use a digital photo frame as an example to illustrate our development process which combines the UPES design process with UML-RT notations.

6.1. System description and external events

A digital photo frame is designed to play photos. These photos can be played forward or backward by who's interesting, or changed the automatic play speed. It also has some additional functions, such as alarm clock...etc. The user can use USB port to connect with the digital photo frame to upload and download photos.

The main features of the DPF are listed below:

- A USB port which connect the external USB storage to save the photos.
- A Touch-screen. The user can control the software directly. The photos can be shown on this display.
- A set of buttons. We also use them to control the system.
- An alarm clock. It is always show on the display. The user can set the current time through the buttons or touch-screen.
- A sound speaker. We use it when the system generated an exception from user operation or make an alarm of setting time.
- External power supply. Then we won't consider that we must change the battery when the power of battery low.

An embedded system constantly interacts with its environment through the events. The following table shows all the external event can occur in our system. The direction of an event can be "In", from the environment to the system, or "Out" in vise versa. The arrival pattern - AP can be Periodic, working in the fixed time, or Episodic, the event generated form user or exception. The response time sets an upper bound for the system response actions.

Table 1: External event list

	Event	Direction	AP	Res.
1	A second passes	*In*	*P*	*0.5s*
2	User press button	*In*	*E*	*0.5s*
3	Alarm clock	*In*	*E*	*0.5s*

6.2. Uses cases

Use case diagram is used to describe the functionality of the system from the user's view. In figure 3, there are four different use cases in our system. "Play photos automatic", photos can be shown on the display automatically. The user also sets the speed to 1x, 2x, 4x. “Play photos by hand”, photos can be played forward or backward by user control. “Set clock time”, the user can set the clock time through the buttons or the touch-screen. Finally, the functionality of “Set alarm time” is that the user can set an indicated time to awake by who’s self.

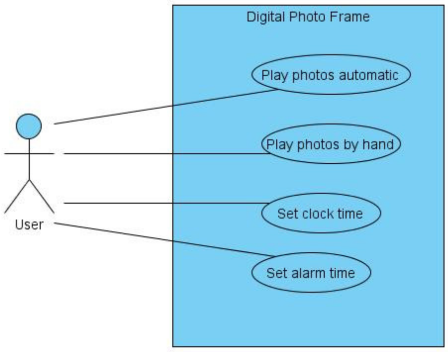

Figure 3: Use case diagram

6.3. Scenario

The scenarios specify the interactions between the external actors (the users) with the system. A sequence diagram for a successful “Play photos automatically” scenario is shown in Figure 4. The user sends a request “playPictures” to system through the touch-screen, then the system response a file management process to display the user he/she can choose the file location and select the files which he/she wants to show. When the user selected the files by buttons, the selected photos will be shown on touch-screen.

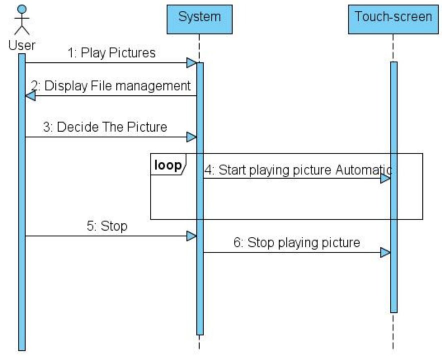

Figure 4: Play photo automatically scenario

6.4. Modeling the hardware structure and behavior

In the case, we select the capsule collaboration diagram of UML-RT to describe the structure of the system; and then we use statechart to present the behavior of the system. Figure 5 is the capsule collaboration diagram of the PDF system. The diagram indicates the DPF system realized of the four sub-capsules of “touch-screen”, “storage”, “buttons”, and “speaker”.

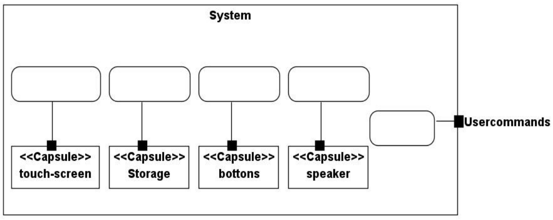

Figure 5: Capsule collaboration diagram of DPF

The rectangle specifies the APIs of the peripherals. It connected with the capsules through the ports and connectors. When the user makes a command to the system, the system will invoke the corresponding APIs of peripherals to service the request.

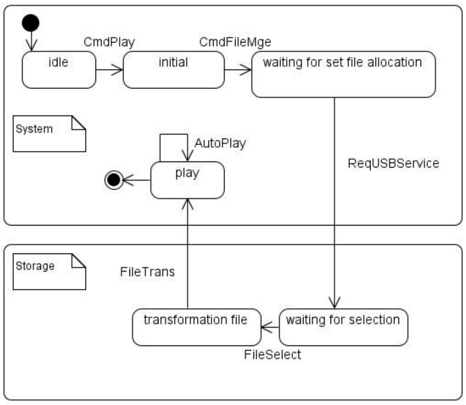

Figure 6: Statechart of DPF

Just like the state diagram defined by UML, Figure 6 shows the behavior between the DPF system and the USB storage by statechart.

6.5. Modeling the mapping model

Finally, in the meet-in-the-middle [11] phase, we iterative mapping and testing the mapping model, to confirm the mapping of Application and Platform model is correct. When the embedded system structure is decided, we use the deployment diagram to represent the result (see Figure 7). There have the processor, ram, and peripherals actives in this system. The signal transmits through the bus.

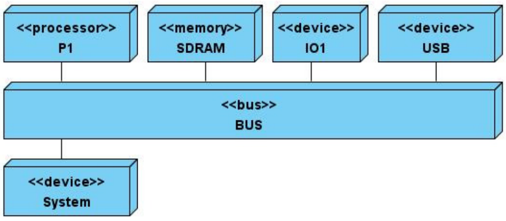

Figure 7: Deployment diagram of DPF

6.6. Implementation

When the mapping model has been completely developed and tested, it can be deployed to the target board. In the Figure 8, we use the DMA-2440 development platform to implement the DPF system.

The main processor of DMA-2440 is Samsung S3C2440A, 400MHz, and the main memory is 64Mbytes. It relies on the Sharp LCD display, 240x320 pixels. The sound is played through the speaker by using a sound chip UDA1341, and the control activity realized by using the on board buttons. The USB port can used to increase the capacity.

We have chosen the Qt-Embedded [8] programming language for implementing our system. Qt-Embedded is a C++ like language which work on the embedded Linux. The signal-slot mechanism of Qt-Embedded is helpful to programmer.

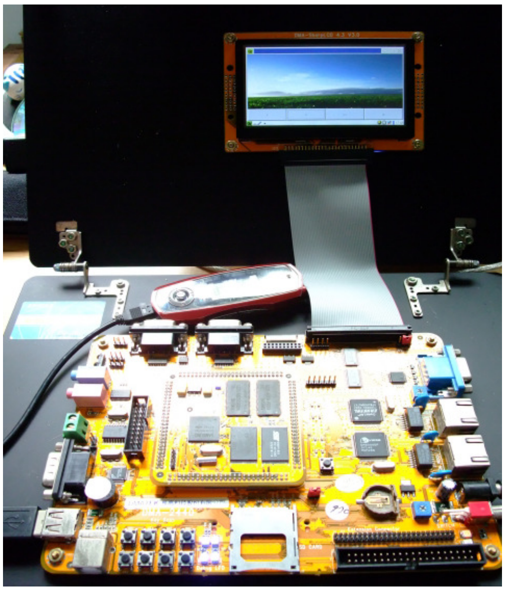

Figure 8: DMATEK DMA-2440 development platform

7. Conclusion

We have proposed an approach to model the embedded software using the UPES process and the UML-RT notations. In the Application model, we defined the requirements and external events which active between the environment and system, and then we according to the UP process to develop the digital photo frame and use the use case diagram and sequence diagram to describe the use case. In the Platform model, the capsule collaboration diagram and statechart are used to describe the structure of the system. Finally, we construct the Embedded System Platform model by deployment diagram.

Our future research plan will focus on the following tasks:

- To investigate the issues of providing a connection between the DPF and Internet photo album, such as wretch web site.
- To integrate the goal perspective for future improving the proposed approach to address non-functional issues, such as performance and reliability etc.

8. Reference

[1] J. Arlow and I. Neustadt, *UML and Unified Process*. Addison Wesley, 2002.

[2] Gerd Frick, Barbara Scherrer, and Klaus D. M¨uller-Glaser," Designing the Software Architecture of an Embedded System with UML 2.0," FZI Forschungszentrum Informatik, white papers, October 2004.

[3] P. N. Green, and M.D. Edwards, "The modelling of embedded systems using HASoC," *Design, Automation and Test in Europe Conference and Exhibition*, 2002. Proceedings, 2002.

[4] P. N. Green, Martyn Edwards, and Salah Essa. HASoC - Towards a New Method for System-on-a-Chip Development. *Design Automation for Embedded Systems*, vol. 6, no. 4, July, 2002.

[5] Sang-Uk Jeon, Jang-Eui Hong, and Doo-Hwan Bae. Interaction-based behavior modeling of embedded software using UML 2.0. *Object and Component-Oriented Real-Time Distributed Computing*, 2006.

[6] OMG. The Unified Modeling Language (UML). http://www.uml.org.

[7] Ivan P. Paltor, and Johan Lilius, "Digital Sound Recorder: A Case Study On

Designing Embedded Systems Using the UML Notation," Turku Centre for Computer Science, Technical Report: TUCS-TR-234, January, 1999.

[8] Qt-Embedded. http://trolltech.com/solutions/industrial-embedded-development

[9] Elvinia Riccobene, Partizia Scandurra, Akberto Rosti, and Sara Bocchio, "Designing a Unified Process for Embedded Systems," in the *Proceedings of 4th International Workshop on Model-based Methodologies for Pervasive and Embedded Software (MOMPES'07)*, August, 2007.

[10] B. Selic, and J. Rumbaugh, "Using UML for Modeling Complex Real-Time Systems," white paper, Rational, March 11, 1998.

[11] A. S. Vincentelli, Defining Platform-based Design. *EEDesign of EETimes*, February, 2002.

[12] Qiang Zhu, Akio Matsuda, Shinya Kuwamura, Tsuneo Nakata, and Minoru Shoji," An object-oriented design process for system-on-chip using UML," *Proceedings of the 15th international symposium on System Synthesis*, 2002.

[13] Bas Graaf, Marco Lormans, and Hans Toetenel. Embedded software engineering: the state of the practice. *IEEE Software*, 20(6):61-69, 2003.