International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:1, No:12, 2007

A Framework for Product Development Process including HW and SW Components

Namchul Do, and Gyeongseok Chae

***Abstract*—**This paper proposes a framework for product development including hardware and software components. It provides separation of hardware dependent software, modifications of current product development process, and integration of software modules with existing product configuration models and assembly product structures. In order to decide the dependent software, the framework considers product configuration modules and engineering changes of associated software and hardware components. In order to support efficient integration of the two different hardware and software development, a modified product development process is proposed. The process integrates the dependent software development into product development through the interchanges of specific product information. By using existing product data models in Product Data Management (PDM), the framework represents software as modules for product configurations and software parts for product structure. The framework is applied to development of a robot system in order to show its effectiveness.

***Keywords*—**HW and SW Development Integration, Product Development with Software.

I. Introduction

CURRENT product development includes a substantial amount of software components in order to provide flexible features satisfying various customer needs. As an example, electronic components in automobiles including their control software hold 20% of the total cost. The cost of recent hybrid cars increases to 47% of the total cost. The share of software in the electric components hold 10% in 2005 and it will increase to 20% in 2015. These statistics makes automobile industry consider the software as an independent and critical component [1]. The introduction of software into products has widely spread into various industries.

It is difficult to develop product integrating hardware and software components since their development approaches are very different from each other. The computer-aided development tools for the two areas also have developed independently and are difficult to be connected for the integrated product development. However, as development of product associated with software becomes more general and complicated, it is important for companies to integrate computer-aided hardware and software development tools.

Software engineers propose integrated frameworks for Software Configuration Management (SCM) for software development and Product Data Management (PDM) for general product development [2,3,4]. Since SCM experts find more advanced and general product evolution control in PDM, which is a common concept for both the areas, they can propose frameworks that can share the common concept to support both software and hardware product development. The frameworks supported by SCM experts provide integrated product data models and system architecture. However, they can not answer the questions of the basic principles of the integration and considerations of complex and various interactions between the two areas during the product development processes.

This paper provides an integrated framework for supporting the product development with software components on the view of PDM experts. Objectives of the framework include providing basic principles that can classify the dependent software to be integrated to hardware products from the whole software development. Another objective is how to integrate the software components to existing product configurations and assembly structures for hardware products. The framework is consists of the principles, an integrated product data model, and description about product development process supporting associated software component. The framework is applied to development of a network-based robot system in order to show its effectiveness.

In the following, we first describe a framework for integrated product development including both hardware and software components. Thereafter, we discuss an application of the framework to developing a robot system. We conclude this paper with further research topics.

II. A Framework for Product Development including HW and SW Components

A. Basic Principles

The proposed framework is based on the following principles:

1) The framework integrates SCM features into PDM environment.
2) The framework separates the dependent software from the whole software development for its integration to hardware development.

Manuscript received Febrary 15, 2006.

Namchul Do is with Div. of Industrial and Systems Engineering, Gyoeng Sang National University, 900, Kajwa-dong Jinju City, Geongnam, Korea 660-701 (phone: +82 55 751 6214; fax: +82 55 762 6599; e-mail: dnc@gnu.ac.kr).

Kyeongseok Chae is with Div. of Industrial and Systems Engineering, Gyoeng Sang National University, 900, Kajwa-dong Jinju City, Geongnam, Korea 660-701 (e-mail: pro_cks@hotmail.com).

3) The framework consists of product development process and product data model.

The first principle shows our integration strategy is a PDM centered approach. PDM has richer set of product revolution management functions than SCM, since it has longer application history and common standards such as ISO STEP [2]. Another factor affects the principle is existing research on software product families or product lines [5,6] in software engineering. Software product family concerns combination of software modules to satisfy various requirements from different customers. It imports its concept from product configuration management in PDM. Dahlqvist et al. [3] also proposes extensions of product structure in PDM to integrate software parts.

The second principle differentiates the dependent software (on hardware) from general software. Current products with software consist of various software components including embedded software, firmware, operating systems, and client server applications. Existing approach does not classify software to be integrated with hardware components. If there is no classification of the software, the integration is impossible since there are too many kinds of software to be integrated into product development process. The dependant software proposed in the framework enables design engineers to efficiently integrate the software component into product development.

The framework also considers modification of existing product development process, which can support the integrated software development. There are many mismatches between hardware and software development and the proposed framework considers the differences and how to resolve the problems within an integrated product development process.

B. Integrated Product Development Process

Fig. 1 depicts the process in the framework. The process shows phases and information flows for hardware, software, and the dependant software development.

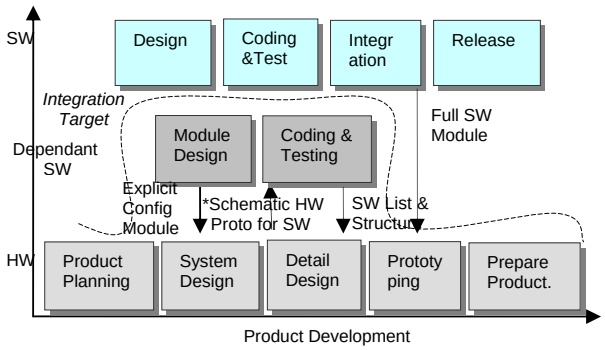


Fig. 1 Product Development Process in the Proposed Framework

The dependant software is the software that is tightly related to hardware components. We classify the dependent software from general software with the following criteria:

1) The dependent software activates hardware components such as motors or actuators or processes input data from hardware components such as sensors.
2) The dependent software should support software modules for product configuration models.
3) Since the dependent software is tightly associated with hardware parts, the changes of associated parts can affect the change of the software.

General product development process consists of the 5 phases in Fig. 1. The second phase, System Design phase, produces product configuration model that consists of pre-defined modules in order to generate large number of product variations. For the product configurations with software components, software modules should be provided for the associated hardware product modules in the configurations.

In the 3rd phase, Detail Design phase, design engineers generate product specifications including part shape, material, or manufacturing method. During Detail Design phase, programmers for the dependent software build and test software that control hardware parts. For the coding and testing, they need a specific hardware system that can provide input and output for their programs. However, hardware engineers usually do not complete their design at this stage. They can provide only schematic prototype hardware for the programmers to build and test the software components. At the end of Detail Design phase, the programmers should provide a software list and its structure that correspond to the hardware components. The software parts and the product structure are managed in PDM for the integrated product development.

C. Integrated Product Data Model

Fig. 2 shows the product data model in the proposed framework. The model is a general product data model introduced by many PDM systems. In the model, product configurations are composed with pre-defined modules, Options, and the modules consists of several physical assembly parts. The parts, Part object in Fig. 2, can be associated with other parts through the *consists of* relationship which creates product structures. Parts can have engineering documents which represents different engineering aspects of the parts. The documents are managed by an evolution model in order to keep the track of design alternatives

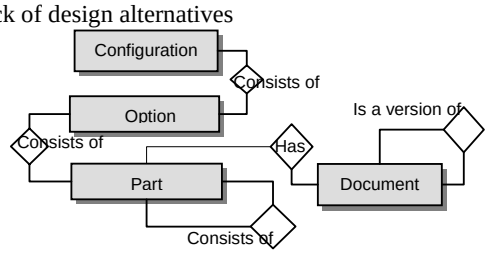


Fig. 2 Product Data Model in the Framework

Representation of Integrated Product Configuration

Current PDM introduces implicit product configuration model that provides only modules (see Options in Fig. 2) and their composition rules without exact specifications for each

product configuration. The implicit product configuration model can allow engineers or customers to compose various product configurations. The dependent software also supports modules of software for the product configuration model. Although each composition of the modules (an implicit product configuration) can consist of numbers of software modules, the framework provides only one software module for each product configuration. It is because a software module is usually provided as a complied or merged file for the end configuration. Since the software modules in each product configuration supports only one specific product configuration, their model can be classified into explicit product configuration models.

Fig. 3 shows an example of product structure based on the product data model of the framework. In the example, product configuration Config A and Config B consists of hardware Options A, B and B, C respectively. On the other hand, the framework supports only one software option for each product configuration (see Option SW A and Option SW B for Config A and Config B in Fig. 3).

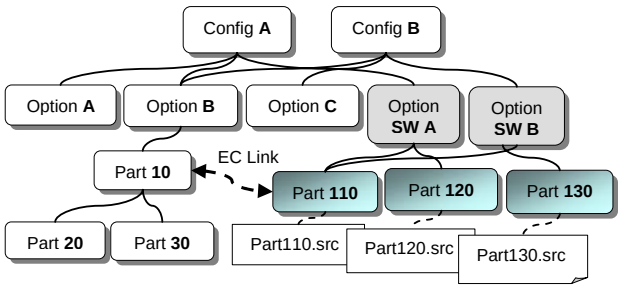


Fig. 3 An Example of SW Options for Integrated Product Configurations

Realization of Software Parts

The meta-information of dependent software is represented as a software part and its source code is managed by the document objects associated with the part. Since the software part shares the same data structure and properties as the hardware part, it can build product structures and share software parts and their source code (see Part110 shared by two Options in Fig. 3). The tightly related software and hardware parts are linked with Engineering Change (EC) links through which a change of a hardware part can be propagated to the related software part.

III. Application of the Framework

A. Development of a Network-based Robot System

The proposed framework is applied to development of a network-based robot system. The prototype robot system for research purpose is a mobile robot based on wireless Internet. Users can remotely control the robot and change its behavior by downloading new control programs through the network. Fig. 4 shows the architecture of robot system and the picture of the mobile robot.

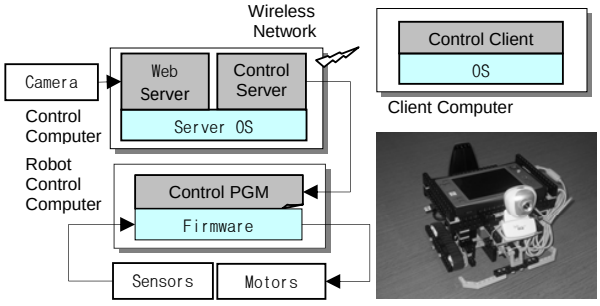


Fig. 4 The Architecture of Robot System and Picture of Mobile Robot

In this development project, Dassault Systems SmarTeam V5 R14 and Microsoft Visual Source Safe 6.0 are used as main PDM and SCM respectively. Various software modules and programs are developed by using different program languages and software development environment.

B. Application of the Framework Separation of Dependent SW

The robot system needs various kinds of programs. It includes operating systems for control server and client, firmware for robot control computer, network programs for the remote robot, a server program for sending moving images, and robot control programs for robot behaviors. Among the programs, we classify the robot control program as the dependent software using the concept proposed in the framework. The robot control program runs on a specific robot control computer to realize the automotive behaviors of the robot by processing input signals or activating motors. Therefore the program is highly related to hardware components such as sensors and motors. The robot control program is abstracted to the software parts and they are managed by the PDM. Other software including client server programs for the remote control are managed by the SCM tool.

SW Options for Product Configuration

Fig. 5 depicts the product configurations of the robot system. The robot system provides two product configurations, Argos Type 1S (with Single Bumper Touch Sensor) and Argos Type 1D (with Double Bumper Touch Sensor). The two configurations share 4 from 5 hardware options. The framework provides individual software Options for each configuration (see Software Single Bumper and Software Double Bumper Options in Fig. 5). The explicit configuration model for software options support software engineers to easily develop, compile, and test software modules for each configuration.

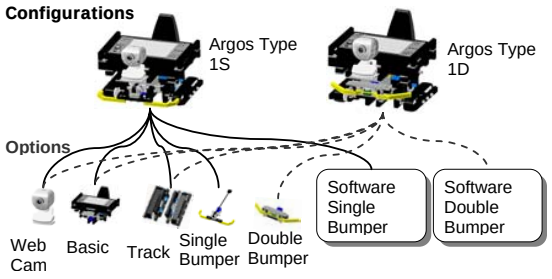
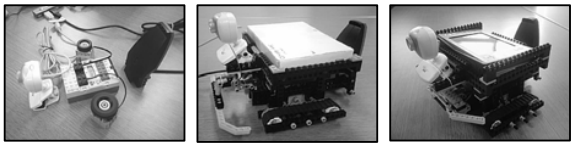


Fig. 5 The Product Configuration Model of the Robot System

Schematic HW Prototype

At the early stage of Detail Design phase, as the framework described, a schematic hardware prototype was provided by hardware engineers to programmers in order to develop the robot control program. Fig. 6 shows the picture of the schematic prototype (see Fig. 6a) and other prototypes for Prototyping phase in the product development process.



(a) (b) (c)

Fig. 6 Schematic HW Prototype (a) and other prototypes for Prototyping phase (b, c)

SW Parts and Product Structure

At the end of the Detailed Design phase, programmers provide a software part list and associated product structure. Fig. 7 shows the software part structure used in the example. Using the product structure, the software parts share other software parts and their source codes. The documents management feature of PDM enables the programmers to manage the concurrency and revolution of the source codes. EC links also support programmers to review the related software parts when ECs occur on the hardware parts.

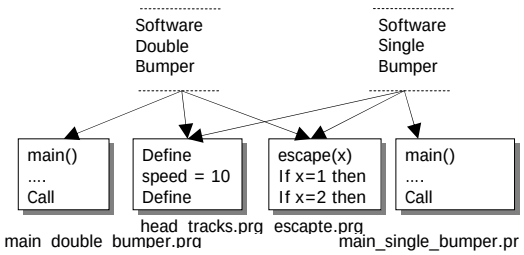


Fig. 7 An Example Software Product Structure

IV. Conclusion

This paper proposes a framework for product development including hardware and software components. It provides concepts for the dependent software for integrated product development with software. In order to integrate the software into product configurations and assembly structures, the framework abstracts the software into software parts. A modified product development process is also introduced into the framework in order to manage the integration of the dependent software. Currently we are developing an extension of PDM system that implements SCM features such as source code comparisons.

References

[1] B. S. Choi and D. K. Bok, “Trends of Convergence in Automobile and IT,” (in Korean) *SERI Economic Focus*, No 65, 2005.
[2] J. Estublier, J-M Favre and P. Morat, “Toward SCM/PDM Integration,” *System Configuration Management, SCM-8*, Lecture Notes in Computer Science 1439, Springer, pp. 75-94, 1999.
[3] A. P. Dahlqvis, I. Crnkovic and M. Larsson, “Managing Complex Systems – Challenges for PDM and SCM,” *Software Configuration Management, SCM-10 23th ICSE*, Toronto, Canada, 2001.
[4] I. Crnkovic, U. Asklund, and A. P. Dahlqvist, *Implementing and Integrating Product Data Management and Software Configuration Management*. Artech House, 2003.
[5] T. Mannisto, T. Soininen and R. Sulonen, “Configurable Software Product Families,” *ECAI 2000 Configuration Workshop*, Berlin, Germany, 2000.
[6] T. Krebs, K. Wolter and L. Hotz, “Model-based Configuration Support for Product Derivation in Software Product Families,” *PuK 2005*, 2005.