THE INTEGRATED METHOD FOR A SYSTEMATIC APPROACH IN DEVELOPING PRECISION FARMING SOFTWARE PRODUCT LINE ARCHITECTURE

NORAZIAN BINTI MOHAMAD HAMDAN

UNIVERSITI TEKNOLOGI MALAYSIA

THE INTEGRATED METHOD FOR A SYSTEMATIC APPROACH IN DEVELOPING PRECISION FARMING SOFTWARE PRODUCT LINE ARCHITECTURE

NORAZIAN BINTI MOHAMAD HAMDAN

A dissertation submitted in partial fulfillment of the
requirements for the award of the degree of
Master of Science (Computer Science)

Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia

JUNE 2012

This dissertation is dedicated especially to my beloved mother and late father (Fatimah bt. Mohammad and Mohd. Hamdan bin Edi) and also not forgetting my beloved siblings for their endless supports and encouragements.

ACKNOWLEDGEMENT

First and foremost, I would like to express my utmost gratitude to Allah s.w.t for endless blessings and given me strength to during the development of this research until it has completed.

I am also heartily grateful and thankful to my supervisor **Dr. Dayang Norhayati Abang Jawawi** for her constant support during my study at UTM. She inspired me greatly to work in this project. Her motivations and guidance have contributed tremendously in completing the research. I have learned a lot from her and I am fortunate to have her as my mentor and supervisor.

My sincere appreciation also extends to all my fellow classmates and friends for willingness to share knowledge and provide assistance when needed. I am also grateful for the love and endless support from my family members especially my mother.

Besides, I would like to thank the authority of Universiti Teknologi Malaysia (UTM) for providing me with a good environment and facilities during my stay in UTM.

ABSTRACT

Software complexity has always been an issue in software development, especially for larger system with innovative functionalities. One solution to reduce the problem of software complexity is by applying software reuse method. Nowadays, Software Product Line is an emerging paradigm for handling software development for software reuse. Software Product Line usually associated with using feature-oriented development methods. Although there are many methods that existed for developing Software Product Line, there are no concrete descriptions on systematic processes that the methods used. Therefore, in this research, an integration of three feature-oriented Software Product Line methods is proposed for developing software architecture. The methods are Feature-Oriented Domain Analysis (FODA), Feature-Architecture Mapping (FArM), and Feature Dependency Analysis (FDA). The case study selected for this research is Precision Farming application based on Wireless Sensor Network. The proposed integrated method provides systematic processes with detailed guidelines for designing software architecture. The software architecture for the case study is developed using the proposed integrated method. The proposed integrated method is evaluated with the initial FArM and has higher benefits because the integrated method includes domain analysis and enhanced mechanism for handling feature dependencies. When compared to Feature-Oriented Reuse Method (FORM), there are fewer differences, but the integrated method provides stronger mapping from features to architectures because it uses FArM method.

ABSTRAK

Kerumitan perisian sentiasa menjadi isu dalam pembangunan perisian, terutamanya bagi sistem dengan fungsian inovatif yang lebih besar. Satu penyelesaian untuk mengurangkan masalah kerumitan perisian adalah dengan menggunakan perisian kaedah penggunaan semula. Pada masa kini, Perisian Talian Produk adalah suatu paradigma baru untuk mengendalikan pembangunan perisian untuk perisian guna semula. Perisian Talian Produk biasanya dikaitkan dengan menggunakan kaedah pembangunan berorientasikan ciri. Walaupun terdapat banyak kaedah yang wujud untuk membangunkan Talian Produk Perisian, tidak terdapat sebarang penerangan konkrit pada proses sistematik bagi kaedah yang digunakan. Oleh itu, dalam kajian ini, integrasi tiga berorientasikan ciri perisian kaedah Talian Produk dicadangkan untuk membangunkan senibina perisian. Kaedah-kaendahnya ialah *Feature-Oriented Domain Analysis (FODA), Feature-Architecture Mapping (FArM),* dan *Feature Dependency Analysis (FDA)*. Kajian kes yang dipilih untuk kajian ini adalah kaedah pertanian terperinci berdasarkan rangkaian pengesan tanpa wayar. Cadangan kaedah yang bersepadu menyediakan proses sistematik dengan garis panduan terperinci untuk reka bentuk senibina perisian. Senibina perisian untuk kajian kes ini dibangunkan dengan menggunakan kaedah bersepadu yang dicadangkan. Kaedah bersepadu yang dicadangkan telah dinilai dengan kaedah FArM dan mempunyai kelebihan yang tinggi kerana kaedah bersepadu mengandungi analisis domain dan mekanisma dipertingkatkan untuk mengendalikan kebergantungan ciri. Apabila dibandingkan kaedah *Feature-Oriented Reuse Method (FORM),* terdapat sedikit perbezaan, tetapi kaedah bersepadu menyediakan pemetaan yang lebih kukuh dari ciri-ciri ke senibina kerana ia menggunakan kaedah *FArM*.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AP	*Access Point*
DGPS	*Differential Global Positioning System*
DSS	*Decision Support System*
EC	*Electrical Conductivity*
ECS	*Elevator Control System*
FArM	*Feature-Architecture Mapping*
FDA	*Feature Dependency Analysis*
FeatuRSEB	*Featured RSEB*
FMIS	*Farm Management Information System*
FODA	*Feature-Oriented Domain Analysis*
FORM	*Feature-Oriented Reuse Method*
GIS	*Graphical Information System*
GPRS	*General Packet Radio Service*
GPS	*Global Positioning System*
HyperFeatuRSEB	*Hyper Feature RSEB*
LAN	*Local Area Network*
LGIS	*LG Industrial Systems CO. Ltd*
MEMS	*Micro-Electro-Mechanical Systems*
MOSAICo	*Modelo de Objetos de Sistemas Abertos de Informação de Campo*
NIMSAD	*Normative Information Model-Based Systems Analysis and Design*
OMG	*Object Management Group*
OPAIMS	*Open Architecture Precision Agriculture Information Monitoring System*
PCB	*Printed Circuit Board*
PDA	*Personal Digital Assistant*

PF	*Precision Farming*
RF	*Radio Frequency*
RSEB	*Reuse-Driven Software-Engineering Business*
SOSD	*Service-Oriented Software Development*
SPEM	*Software Process Engineering Meta-Model*
SPL	*Software Product Line*
SPLE	*Software Product Line Engineering*
SDSS	*Spatial Decision Support System*
TCP/IP	*Transmission Control Protocol/Internet Protocol*
UDP	*User Datagram Protocol*
WLAN	*Wireless LAN*
WSAN	*Wireless Sensor/Actuator Networks*
WSN	*Wireless Sensor Networks*

CHAPTER 1

OVERVIEW

1.1 Introduction

Towards the 21st century, information technologies have rapidly advancing and have since been applied to many fields, including agriculture. In the past, the agricultural-related activities were performed in a traditional way involving human labors to operate equipments and machines. Farmers need to constantly visit the crop field to monitor the crop conditions and the data were measured manually crop by crop. Decisions on harvesting as well as the suitable amount of fertilisers and pesticides can only be made after gathering enough information from the crop field. The traditional way was time consuming especially if it involves monitoring large-sized crop fields. Fortunately with the advancement of information technologies, the agricultural activities were modernised by enabling automation to replace manual operations. Hardware devices like sensors are deployed on the crop field to gather and measure data, and software system is used to process the data for decision-making. As a result, the needs for human labors have been decreased and able to reduce the time it takes for data collection. This new method is known as Precision Farming (PF), or sometimes called as Precision Agriculture.

Ever since PF has been introduced, there are many definitions and concepts of PF that have been described by researchers and practitioners. The most cited and used definition is by Robert *et al.* (1994), who have proposed three "R", which are the Right time, the Right amount and the Right place. Another two "R" was added to the list by the International Plant Nutrition Institute and Khosla (2008). The former added the Right source, and the latter added the Right manner. Other definition of PF

are by Pierce and Nowak (1999), who defined PF as the application of technologies and ideas to manage agricultural spatial and temporal variability in order to improve crop production and environmental quality.

PF combines two different field of expertise. First is the information technology field that includes the implementation of Wireless Sensor Networks (WSN), Global Positioning System (GPS), Graphical Information System (GIS), and Decision Support System (DSS). Among the devices used in PF are remote sensors, yield monitors, yield mappings, variable rate technologies, and such (Sudduth, 1999). Second is based on the accumulated farmers' experiences and vast knowledge on fertilisers, pesticides, and variations in nutrient availability, crop and soil properties, root growth, and other important factors.

PF is a site-specific management, which means variations are monitored and studied at specific field which are divided into smaller areas. In order to allow for monitoring and controlling of different areas in a crop field, the setup of WSN is necessary. WSN integrates latest technologies like radio frequency (RF) transceivers, sensors, microcontrollers, and power sources (Wang *et al.*, 2006). The implementation of WSN technology in PF helps to increase efficiency in real-time processing for environmental controlling and monitoring of factors like temperature, water, and soil chemistry, which is crucial in PF to maintain the ideal field environment of a crop. WSN is seen as a reliable technology with high potentials at present as well as in the future. As a matter of fact, there are several published papers concerning PF application using WSN. For example, Li *et al.* (2008) discussed on-field installation of WSN based greenhouse that employed management strategy in order to achieve such cost-effective and ecological friendly greenhouse management; Pierce and Elliott (2008) described the deployment of regional and on-farm sensor network in for two PF applications in Washington State; Camilli *et al.* (2007) conducted a simulation which is aimed to prove the utility of WSN in PF; and López *et al.* (2010) and Riquelme *et al.* (2009) constructed a WSN prototype for measurement of data on soil and environmental features for vegetable farms irrigation.

Software system is also an important element in PF to allow the processing of collected data and to assist in decision-making. PF software system consists of modules for data acquisition, data collection, recording of data, sophisticated data processing, decision-making, complex data computations, and repository management. PF software system is constructed based on the software architecture that is designed according to the given PF software development requirements. Software architecture is a systematic structure that contains software components and description of their external properties and definition of relationships among them (Bass *et al.*, 1998). The architecture determines two important aspects of the system; what it is and what it does. In other words, the software architectures represent the main body of a system that includes rich information like logical flows, protocols, interfaces, and processes.

The issue of complexity is quite common in computing field since a decade ago, especially in software development. Among the causes that contribute to software development complexity are customer's demands for innovative system functionalities, many kinds of different platforms created by vendors, and the ever changing requirements by the customers. Therefore, the code size and error rates will increase and it will become difficult to maintain.

Reusability can be an appropriate strategy to solve the complexity issue mentioned previously. The term reusability refers to the properties of software that specifies its possibility of reuse (Frakes and Kang, 2005). Software reuse is described as the process of developing new software by using existing software or software knowledge (Pohl *et al.*, 2005). Software Product Line Engineering (SPLE) is one of emerging paradigm that promotes reusability of software assets like components, architectures, designs, data, and modules (Pohl *et al.*, 2005). SPLE in software development aims to produce software at lower costs and in shorter time without neglecting the software quality. The two important approaches in SPLE are extracting commonalities and managing variability in the software system (Gomaa,

2005). Among benefits that can be obtained from software reuse are gains in productivity, gains in quality, and gains in development schedule (Mili *et al.*, 2002). The software products that can be reused are executable code, source code, requirements specifications, designs, test data, documentation, and architectures (Mili *et al.*, 2002).

In SPLE, the software reuse approach is applied to the system family. The term "system family" refers to a set of software system with similar features that satisfy specific given requirements and are developed from a set of reusable core components in a predefined way (Pohl *et al.*, 2005). The reusable core components, also known as core assets, contain software components and requirements, analysis, design and test artifacts, are stored in a shared repository. These core assets are made available for reuse by multiple members of the system family, with variability mechanisms. The main concept for reuse in SPLE is to exploit the commonality of core assets and manage their variability.

The SPLE approach is as shown in Figure 1.1. The architecture of the product is shared by every members of the system family. Commonality of the functions that the system family must employ is recognised by similar components that applied to the software architecture. Meanwhile, the variability of functions that the system family may or may not employ is recognised by the integration of different components attached to the variation points of the software architecture.

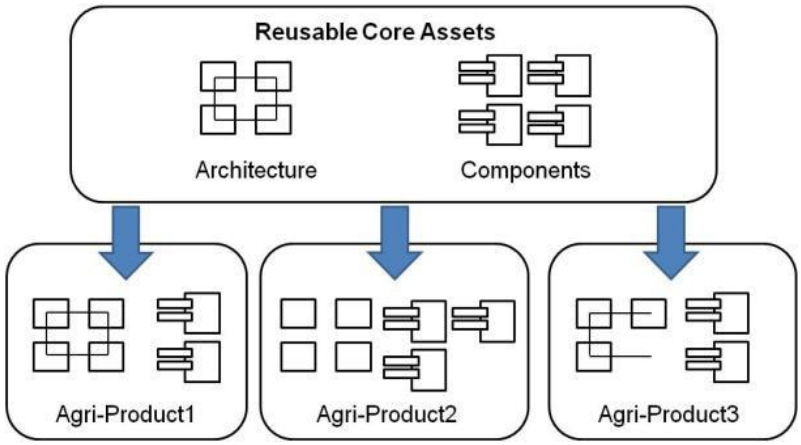

Figure 1.1: Architectural approach reuse in SPL (Fajar *et al.*, 2010)

In SPLE, the most important concept is features that serve as representation of reusable components and requirements of a product line. According to Kang *et al.* (1990), features are known attributes of a system that are identified by the end users. The most used method for analysing a domain using features is the Feature-Oriented Domain Analysis (FODA), in which those features are constructed into a feature tree or feature model (Kang *et al.*, 1990). The feature model is composed of a hierarchy of features, with each branch holds either mandatory, optional, or mutually exclusive conditions. The FODA method will be used to document requirements artefacts for the research. The output will be a feature model of a PF application based on Software Product Line (SPL). In order to map the requirements into architectural components, a new feature-architecture mapping method by Sochos and Matthias (2004) will be used, namely Feature-Architecture Mapping (FArM) method. In order to handle the feature variability of the software components, the FArM method will be integrated with processes in Feature Dependency Analysis (FDA) method.

1.2 Problem Background

In this section, there are two types of problems that will be discussed, which are the domain research problem and technical problem. The domain research

problem is related to the issue of complexity in software development. The technical problem is involving the issues on the SPLE methods that are used for software development.

In the past, building software from scratch may be the only solution in software development. The basic development phases will start from collecting the requirements from customers and stakeholders, analysing them, and then designing the architecture or solution before implementing them into lines of codes. Nowadays, building software from scratch may not be the most convenient way because the software development has becoming more complex. Complexity can increase due to evolution of software functionalities, in which is caused by customer's demands to add more innovative functionalities to the software (Pohl *et al.*, 2005). As the complexity increases, the size of code will also increase. An example is the Windows Operating System, in which in 1991 Windows NT PDK 2 had 1.8 million SLOC (Cusumano and Selby, 1998) and Windows XP had 48 million SLOC (Pohl *et al.*, 2005).

As mentioned in previous section, SPLE approach promotes reusability and can be used to develop software for reuse. This can be a promising approach to handle complexity issue in software development. Although there are many SPLE methods available, not all of the methods contain guidelines and full documentation on the processes involved. Each of them may have their own strengths in some aspects of software development, but in other aspects they can display weaknesses too. There is a comparison done by Matinlassi (2004) that summarised the SPL architecture design methods, which are COPA, FAST, FORM, KobrA, and QADA. The methods were compared using context, user, contents, and validation aspects from Normative Information Model-Based Systems Analysis and Design (NIMSAD) framework. The author concluded that there is no competition among methods, but each of them has a special goal to achieve. The author also highlighted the lack of methods documentation and thereby there are no concrete descriptions of processes involved. The comparison table is shown in Table 1.1.

Table 1.1: Comparison between SPL architecture design methods

Categories/ Methods	Context	User	Contents	Validation
Component-Oriented Platform Architecting (COPA)	• Component-based & architecture-centric • Covers all aspects of SPL	• Users who use the method for developing artifacts or only make use of the outputs • With extensive architectural descriptions, improve communications among various stakeholders • Has a special notation that required learning • Lack of method documentation	• Three main phases: Domain Engineering, Application Engineering & Platform Engineering • Capture variability with graphical language in architectural design • Five views: customer, application, functional, conceptual, realisation	• Strongest industrial experience with applications in large product families • Ensures quality attributes with non-architectural evaluation methods
Family-Oriented Abstraction, Specification, and Translation (FAST)	• Process-driven method • Covers requirements engineering	• Users who use the method for developing artifacts or only make use of the outputs • Use UML description language, but requires extended tool • Documented in extensive manuals	• Two main phases: Domain Engineering & Application Engineering • Support variability in requirements elicitation • No view/viewpoint	• Ensures quality attributes with non-architectural evaluation methods

Feature-Oriented Reuse Method (FORM)	• Feature-oriented method • Covers requirements engineering, architecture, implementation & process	• Academic audience • Gives common language, improve communication between customers & engineers • Has special notation that required learning • Has provided special guidelines	• Two main phases: Domain Engineering & Application Engineering • Support variability in requirements elicitation • Three architectural models (or viewpoints): subsystem, process, module	• Ensures quality attributes with non-architectural evaluation methods
Component-Based Application Development (KobrA)	• Component-based method • Covers requirements engineering, architecture, implementation & process	• Software engineers & designers • Use commonly known standards like UML • Use UML description language, but requires extended tool • Documented in extensive manuals	• Two main phases: Domain Engineering & Application Engineering • Capture variability with graphical language in architectural design • No view/viewpoint	• Ensures quality attributes with non-architectural evaluation methods
Quality-driven Architecture Design and quality Analysis (QADA)	• Quality-driven & architecture-centric approach • Represents interface between requirements engineering & architecture design, but cannot be considered as systematic approach to gather & analyse requirements	• Users who use the method for developing artifacts or only make use of the outputs • Use commonly known standards like IEEE-Std-1471-2000 • Use UML description language, but requires extended tool • Lack of method documentation	• Three main phases: Context Analysis, Conceptual & Concrete Design, Quality Evaluation • Capture variability with graphical language in architectural design • Four viewpoint: structural, behaviour, deployment, development	• Quality of design is validated with a scenario-based evaluation in two methods: conceptual & concrete

Feature-oriented approach is one of methods to implement SPLE, which manipulates features of software system and then are plotted into a feature model (Kang *et al*., 1990). The most cited and used feature-oriented method is FODA (Kang *et al*., 1990). The main motive of using FODA is to implement domain analysis. There are detailed guidelines documented by the authors in order to implement domain analysis in a systematic manner (Lee *et al.*, 2002). The guidelines are retrieved based on the authors' experiences with feature modeling on previous projects like electronic bulletin board systems, private branch exchange, and elevator control software. The authors concluded that feature modeling using FODA was insightful, effective, and efficient for discovering commonality and variability of domain applications. Therefore, the FODA method is useful for Analysis phase in software development. In order to continue with Design and Implementation phases, FODA method is extended to FORM (Kang *et al.*, 1998). However, one obvious weakness of FORM is that it does not provide concrete description on processes to map features to architectures. FORM does not focus on providing a clear transition between feature model and architecture, instead only concretising the FODA processes of analysis and design from a marketing perspective.

In 1999, the initial FORM method was used in building the core part of Elevator Control System (ECS) for LG Industrial Systems CO. Ltd. (LGIS) (Choi *et al.*, 1999). The development of ECS using initial FORM method was a success. After the ECS was developed, more changes have occurred in the hardware and software environments, as well as evolution of the control technologies. Unfortunately, major efforts were required for handling the system maintenance. Therefore, to reduce the efforts for system maintenance, another research is conducted on the FORM method (Lee *et al.*, 2000). It is an extended to the previous FORM researches, (Kang *et al.*, 1998) and (Kang *et al.*, 1999). In (Lee *et al.*, 2000), a new ECS architecture was developed. Prior to developing the new architecture, feature analysis was done to the whole family products of ECS to identify modules to be hidden, and ways to handle contextual variations like technology and environment variations. The authors have highlighted that feature analysis is a crucial approach in one of the steps for developing software architecture. By implementing the new architecture, any changes occur on the ECS can be done with minimal efforts.

Feature-Architecture Mapping (FArM) method was introduced by Sochos and Matthias (2004) that claimed to have stronger mapping of feature model to architecture compare to FORM. The mapping is done in two phases, which are Transform Feature Model and Building Reference Architecture phases. Transform Feature Model phase contains four transformations that are done iteratively. Building Reference Architecture phase is implemented in parallel with each transformation in order to derived architectural components. Thereby, FArM method provides more systematic processes and is able to map features to architectures stronger that FORM. FArM involves only in the Design and Implementation phases of software development. The authors suggested the use of FODA method for Analysis phase. It is the best choice to use FODA because it has detailed documentation of processes needed for domain analysis. Although both methods are feature-oriented, some set of guidelines is needed in order to transform the feature model so that it can be used as input for designing architecture using FArM. This is because the feature model built using FODA is in four layers, but the feature model required in FArM is in single hierarchical structure. FArM does not provide processes for transforming four-layer feature model to single layer.

A recent related research is from Fajar *et al.* (2010) that used feature modeling in SPL development for Wireless Sensor/Actuator Network (WSAN) based agriculture systems. Previous researches have been done on WSAN which comprise of application domain like controlling and monitoring of farm, precision irrigation, and managing traceability. The authors have highlighted this as a problem because the applications are developed separately, when instead it is more convenient to implement integration and share the software components. Therefore, the authors proposed developing a WSAN feature model by analysing the commonalities and variability that exist among the application domain (Figure 1.2(a) and (b)). The feature model that the authors have developed can be useful for designing software architecture, and is even possible to be adapted for this research. However, the authors did not provide a complete feature model and only display some parts of the feature model.

Another problem that related to SPL methods is the mechanism used for handling feature variability in software architecture design. FArM introduced switching mechanism (Sochos, 2007) that uses Strategy pattern for handling feature variability. However, the author stated that the mechanism only can handle smaller system with less number of features. FORM, however had introduced Feature Dependency Analysis (FDA) (Lee and Kang, 2004) that uses Factory and Proxy pattern for handling feature variability. FDA can be applied for handling feature variability in larger system with larger number of features.

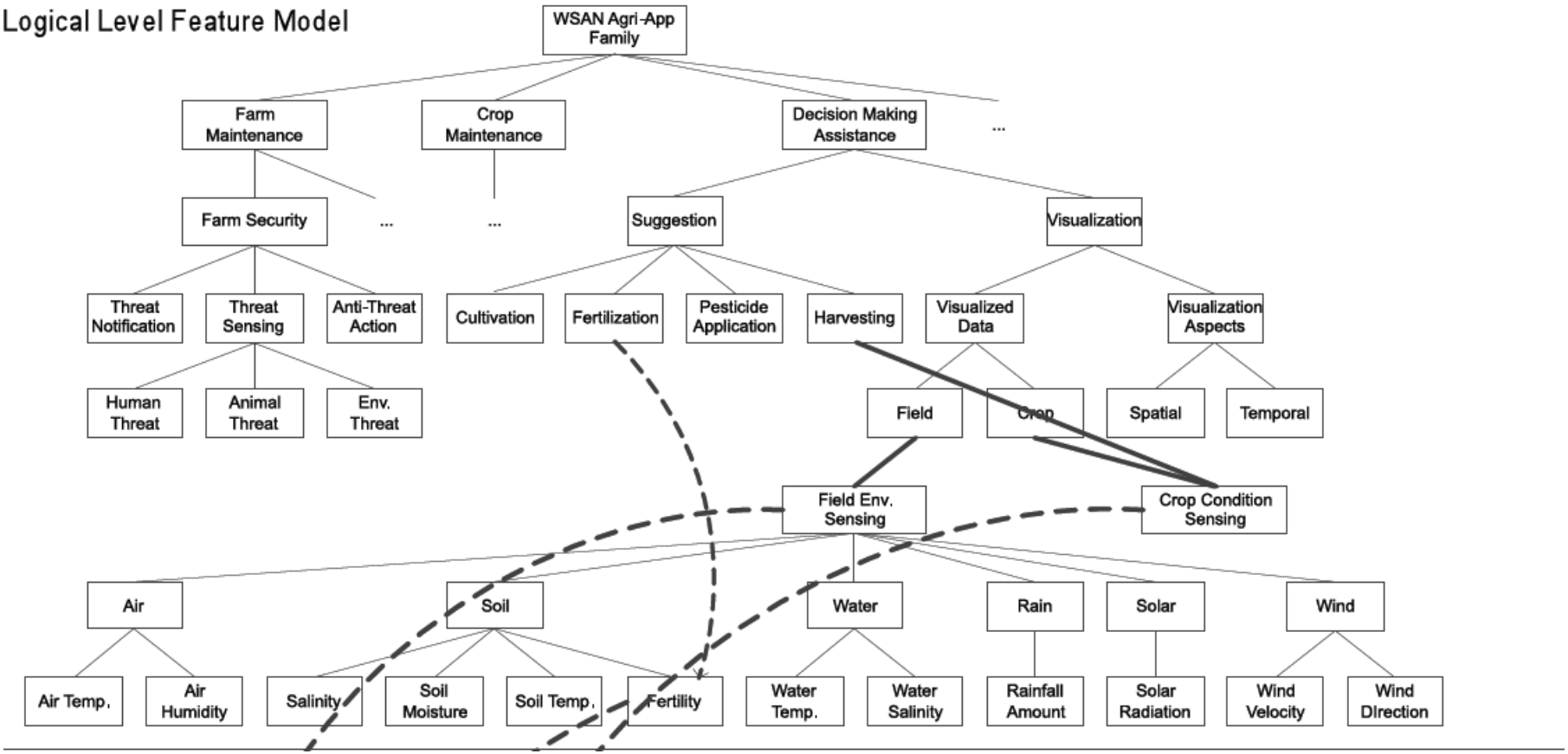

Figure 1.2 (a): Logical feature model of WSAN based agriculture systems (Fajar *et al.*, 2010)

(Refer legend in Figure 1.2(b))

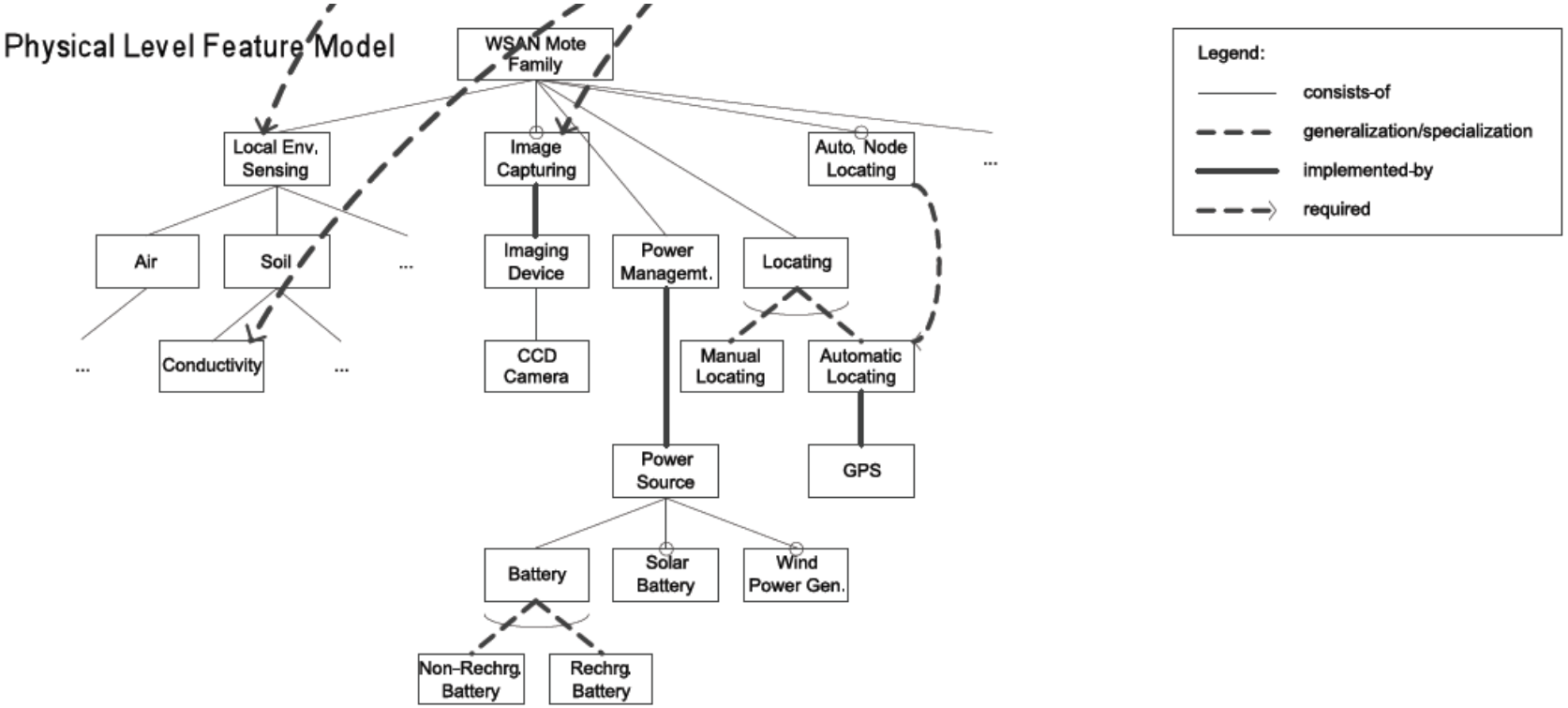

Figure 1.2(b): Physical feature model of WSAN based agriculture systems (Fajar *et al.*, 2010)

1.3 Problem Statement

There are two main problems in this research that need to be solved, which are software complexity and suitable SPL methods to be used. The problem with software complexity is how to handle the issues like changes of requirements during software development, finding the suitable mechanism for handling feature variability, and the artifacts that can be reused. The second problem is in selecting the suitable methods for SPL development. The context of suitable methods refers to the methods that provide step-by-step processes and have sufficient documentation on the methods. There may be some methods that cover all aspects of software development like FORM, but it does not provide step-by-step processes and concrete description on mapping features to architecture. Some methods only cover the analysis aspect of software development like FODA, and some covers design and implementation aspect like FArM. Some methods like FDA only cover the feature variability issues in designing software architecture. The decision to integrate these methods may be a wise decision, but this may require a certain platform or transition phases in order to integrate them.

Therefore, in order to achieve the objectives and to handle the problems that have been discussed in Section 1.2, the study will be carried out to answer the following questions:

1. How to determine which requirements are suitable for developing PF software PL?
2. How to transform the feature model built using FODA so that it can be suitable input for FArM to process?
3. How the integrated FArM and FDA methods can be used for handling feature variability?
4. How to evaluate and validate the proposed integrated FODA, FArM and FDA methods?

1.4 Research Aim

The research is aimed to provide systematic processes and guidelines for developing software architecture. Several existing PF overall system and software architectures that are based on WSN are studied, and then a detailed study is conducted on their elements, features, functionalities, limitations, and abilities. The investigation on software architecture will lead to the process of extracting requirements that will later transformed into features. The features will be represented using feature model using FODA method that will be mapped onto architectural elements using FArM method. FDA method will be used for mapping the feature variability.

1.5 Research Objectives

There are four main objectives that need to be achieved upon the completion of the study.

1. To perform domain analysis by collecting and studying the requirements of PF application and transformed them into features to create a feature model.
2. To integrate FODA and FArM methods by transforming feature model in a way that it can be input for FArM transformations.
3. To enhance the feature variability mechanism in the initial FArM method by integrating it with FDA method.
4. To evaluate and validate the processes in the integrated methods of FODA, FArM, and FDA.

1.6 Research Scope

In order to keep the research within achievable boundary, the conduct of the research is restricted to the following scopes:

1. Analysis and investigation of the study in order to extract requirements are limited to six case studies of existing PF overall system and software architecture.
2. The research only involves the Analysis and Design phases of software development.
3. The use of FODA method is restricted for domain analysis.
4. The FArM method will be used for mapping the feature model onto architectural components.
5. The processes in FDA method that are going to be integrated with FArM method are limited for handling the feature variability.

1.7 Significance of the Study

The study involved research on PF or site-specific management that comprised of multi-disciplinary knowledge on areas like agricultural, information technology development, and engineering innovation. The main purpose of the research is to provide systematic processes and clear guidelines for developing PF software architecture for software reuse. Since PF is a multi-disciplinary approach, the result of the research is contributed to several fields such as agricultural, information technology, and engineering in which specifically aim to promote better practice of software development in PF. It is hoped that the proposed methods can be applied for developing software architecture for software reuse in an application domain. In addition to that, the PF software architecture that is developed in this research can be used as a software reuse artifact for developing applications on PF domain.

1.8 Organisation of Report

Throughout achieving the aims and objectives of the research, the work of research is documented in seven chapters. Chapter One describes the overview of the study by dividing it into sections which consists of introduction of the study, problem background, problem statement, aim of research, research objectives, scope of the study, and significance of the study. In Chapter Two, detailed reviews of past research papers on PF and related topics are included. It involves discussion of theory in PF including its architectures and reviews about software reusability concept. Other than that, Chapter Two includes the literature reviews on existing methods for feature-oriented approach for developing feature models and feature-architecture mapping. This followed by Chapter Three, which contains the research process flowchart and conceptual framework. The research process flowchart shows the flow of the processes that are required in order to complete the research. The conceptual framework shows the steps required to handle the problems that have been stated in Problem Statement and to achieve the research objectives. Chapter Four is the preliminary result of the study, which includes the development of a feature model based on the collected requirements. Chapter Five contains the results of mapping the feature model onto the architectural elements and thus developing the software architecture for reuse. Chapter Six includes the process work definition, which are Requirements Analysis, Analysis, and Design phase. Evaluations of the proposed integrated FODA, FArM, and FDA methods are also included in Chapter 6. Finally, Chapter Seven concludes the results and findings, summarization of the work, and suggestions for future work.

REFERENCES

Atkinson, C., Bayer, J., Bunse, C., Kamsties, E., Laitenberger, O., Laqua, R., Muthig, D., Paech, B., Wüst, J., and Zettel, J., 2002. *Component-Based Product Line Engineering with UML*. Addison-Wesley.

Auernhammer, H., 2001. Precision Farming - the Environmental Challenge. *Elsevier: Computers and Electronics in Agriculture*. 30 (1-3). 31-43.

Baggio, A., 2005. Wireless Sensor Network in Precision Agriculture. *Workshop on Real-World Wireless Sensor Networks*.

Bass, L., Clements, P., and Kazman, R., 1998. *Software Architecture In Practice*. Addison-Wesley.

Boellert, K., 2002. *Object-Oriented Development of Software Product Lines for the Serial Production of Software Systems*. Doctor Philosophy.

Bosch, J., 2000. *Design & Use of Software Architectures - Adopting and Evolving a Product Line Approach.* Addison-Wesley.

Camilli, A., Cugnasca, C. E., Saraiva, A. M., Hirakawa, A. R., and Correa, P. L.P., 2007. From Wireless Sensors to Field Mapping: Anatomy of An Application for Precision Agriculture. *Elsevier: Computers and Electronics in Agriculture*. 58. 25-36.

Charvat, K., Holy, S., Gnip, P., and Sida, A., 2003. Precision Farming Through Internet and Mobile Communication. *Proceedings International Congress IT in Agriculture*.

Chaudhary, D. D., Nayse, S. P., and Waghmare, L. M., 2011. Application of Wireless Sensor Networks for Greenhouse Parameter Control in Precision Agriculture. *International Journal of Wireless & Mobile Networks*. 3.

Choi, B. W., Jang, K. B., Kim, C. H., Wang, K. S., and Kang, K. C., 1999. Development of Software for the Hard Real-Time Controller using Feature-

Oriented Reuse Method and CASE Tools. *Proceedings of the 1999 IEEE International Symposium on Computer Aided Control System Design*.

Cox, S., 2002. Information Technology: The Global Key to Precision Agriculture and Sustainability. *Elsevier: Computers and Electronics in Agriculture*. 36 (2-3). 93-111.

Cusumano, M., and Selby, R., 1998. *Microsoft Secrets: How the World's Most Powerful Software Company Creates Technolog, Shapes Market, and Manages People*.

Fajar, M., Nakanishi, T., Tagashira, S., and Fukuda, A., 2010. Introducing Software Product Line Development for Wireless Sensor/Actuator Network Based Agriculture Systems. *Proceedings of Bogor Agricultural University's Seminars*.

Frakes, W. B., and Kang, K., 2005. Software Reuse Research: Status and Future. *IEEE Transactions on Software Engineering*. 31.

Gomaa, H., 2005. *Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures*. Addison-Wesley

Griss, M., Favaro, J. and d'Alessandro, M., 1998. Integrating Feature Modeling with the RSEB. *Proceedings of the Fifth International Conference on Software Reuse*.

Hwang, J., Shin, C., and Yoe, H., 2010. Study on Agricultural Environment Monitoring Server System Using Wireless Sensor Network. *Selected Papers from the 11th International Workshop of Knowledge Management and Acquisition for Smart Systems and Services*.

Jacobson, I., Christerson, M., Jonsson, P., and Oevergaard, G., 1992. *Object-Oriented Software Engineering: A Use-Case Driven Approach*. Addison-Wesley.

Jayaratna, N., 1994. *Understanding and Evaluating Methodologies: NIMSAD - A Systematic Framework*. McGraw Hill.

Kang, K. C., Cohen, S. G., Hess, J. A., Novak, W. E., and Peterson, A. S., 1990. *Feature-Oriented Domain Analysis (FODA): Feasibility Study*. [Technical Report]. Carnegie Mellon University, Software Engineering Institute. CMU/SEI-90-TR-21.

Kang, K. C., Kim, S., Lee, J., and Lee, K., 1999. Feature-Oriented Engineering of PBX Software for Adaptability and Reusability. *Software-Practice and Experience*.

Kang, K. C., Kim, S., Lee, J., Kim, K., Shin, E., and Huh, M., 1998. FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures. *Annals Of Software Engineering*. 5 (1). 143-168.

Kang, K. C., Lee, J., and Donohoe, P., 2002. Feature-Oriented Product Line Engineering. *IEEE Software*. 19(4). 58-65.

Kheong, L. S., and Jayaratna, N., 2009. Framework for Structuring Learning in Problem-Based Learning.

Khosla, R., 2008. Opening ceremony presentation. *The 9th International Conference on Precision Agriculture*. July 20-23rd.

Larman, C., 2004. *Applying UML and Patterns: Introduction to OOAD & Iterative Development.* (3rd ed.). Prentice Hall, 2004.

Lee, K., and Kang, K. C., 2004. Feature Dependency Analysis for Product Line Component Design. *Software Reuse: Methods, Techniques and Tools*.

Lee, K., Kang, K. C., Chae, W., and Choi, B. W., 2000. Feature-Based Approach to Object-Oriented Engineering of Applications for Reuse. *Software-Practice and Experience*.

Lee, K., Kang, K. C., and Lee J., 2002. Concepts and Guidelines of Feature Modeling for Product Line Software Engineering. *Software Reuse: Methods, Techniques, and Tools*. 2319.

Lewis, K. A., and Bardon, K. S. A., 1998. Computer-Based Informal Environmental Management System for Agriculture. *Environmental Modelling & Software*.

Li, X., Deng, Y., and Ding, L., 2008. Study on Precision Agriculture Monitoring Framework Based on WSN. *International Conference on Anti-Counterfeiting, Security, and Identification (2008 ASID)*. 182-185 .

López, J. A., Soto, F., Iborra, A., Sánchez, P., and Suardíaz, J., 2010. Design and Implementation of a Wireless Sensor Network for Precision Horticulture. *Sensor Applications, Experimentation, and Logistics*. 29.

Mandal, S. K., 2006. Precision Farming Approaches for Small-Farm Agriculture: Scope and Prospect. *Conference Proceedings of Map India 2006 on Agriculture*.

Matinlassi, M., 2004. Comparison of Software Product Line Architecture Design Method: COPA, FAST, FORM, KobrA, and QADA. *26th International Conference on Software Engineering*.

McBratney, A., Whelan, B., Ancev, T., and Bouma, J., 2005. Future Directions of Precision Agriculture. *Precision Agriculture*. 7-23.

Meister, J., 2006. *Product-Driver Development of Software Product Lines Applied on an Example of Analytical Software*. Doctor Philosophy.

Mellon University Carnegie. *Software Product Line*. [Online]. Software Engineering Institute (SEI). Available at: http://www.sei.cmu.edu/productlines/index.html.

Mili, H., Mili, A., Yacoub, S., and Addy, E., 2002. *Reuse-Based Software Engineering: Techniques, Organization, and Controls*. (1st ed.). Wiley-Interscience.

Nikkilä, R., Seilonen, I., and Koskinen, K., 2010. Software Architecture for Farm Management Information Systems in Precision Agriculture. *Computers and Electronics in Agriculture*. 70. 328-336.

OMG, 2008. *System Modeling Language Specification*. [Technical Report]. Object Management Group.

Ossher, H., and Tarr, P., 2001. Multi-Dimensional Separation of Concerns and the Hyperspace Approach. *Software Architectures and Components Technology*. Kluwer Academic Publishers.

Pierce, F.J., and Elliott, T.V., 2008. Regional and On-Farm Wireless Sensor Networks for Agricultural Systems in Eastern Washington. *Elsevier: Computers and Electronics in Agriculture*. 32-43.

Pierce, F. J., and Nowak, P., 1999. Aspects of Precision Agriculture. *Advances in Agronomy*. 1-85.

Pohl, K., Böckle, G., and Linden, F. V. D., 2005. *Software Product Line Engineering: Foundations, Principles, and Techniques*. Springer-Verlag Berlin Heidelberg.

Riquelme, J. A. L., Soto, F., Suardiaz, J, Sanchez, P., and Iborra, A., 2009. Wireless Sensor Networks for Precision Horticulture in Southern Spain. *Computers and Electronics in Precision Agriculture*. 68. 25-35.

Robert, P. C., 1993. Characterisation of Soil Conditions at the Field Level for Soil Specific Management. 60. 57-72.

Robert, P. C., 1999. Precision Agriculture: Research Needs and Status in the USA. *Precision Agriculture, 99 Proceedings of the 2nd European Conference on Precision Agriculture*. Edited by Stafford, J. V. (Sheffield Academic Press Sheffield, UK). 19-33.

Robert, P. C., 2002. Precision Agriculture: A Challenge for Crop Nutrition Management. *Plant and Soil*. 247. 143-149.

Robert, P., Rust, R., and Larson, W., 1994. Site-Specific Management for Agricultural Systems. *Proceedings of the 2nd International Conference on Precision Agriculture*. Madison.

Shibusawa, S., 2001. Precision Farming: Approaches for Small-Scale Farms. *Proc. 2nd IFAC-CIGR Workshop on Intelligent Control for Agricultural Applications*. Bali, Indonesia. 22-27.

Sochos, P., 2007. *The Feature-Architecture Mapping Method for Feature-Oriented Development of Software Product Lines*. Doctor Philosophy.

Sochos, P., and Riebisch, M., 2004. Feature-Oriented Development of Software Product Line: Mapping Feature Models to the Architecture. 3263. 23-42.

Sochos, P., Riebisch, M., and Philippow, I., 2006. The Feature-Architecture Mapping (FArM) Method for Feature-Oriented Development of Software Product Lines.

Stafford, J. V., 2000. Implementing Precision Agriculture in the 21st Century. [Keynote paper]. *Journal of Agricultural Engineering Research*. 76. 267-275.

Streitferdt, D., 2003. *Family-Oriented Requirements Engineering*. Doctor Philosophy.

Sudduth, K. A., 1999. Engineering Technologies for Precision Farming. *International Seminar on Agricultural Machinery Technology for Precision Farming*.

Tian-en, C., Li-ping, C., Yunbin, G., and Yanji, W., 2009. Spatial Decision Support System for Precision Farming Based on GIS Web Service. *International Forum on Information Technology and Applications*.

Tizzei, L. P., and Rubira, C. M. F., *Public Health Complaint Software Product Line*.

Wang, N., Zhang, N., and Wang, M., 2006. Wireless Sensors in Agriculture and Food Industry - Recent Development and Future Perspective. *Computers and Electronics in Agriculture*. 1-14.

Wang, Y., Wang, Y., Xu, L., and Qi, X., 2009. OPAIMS: Open Architecture Precision Agriculture Information Monitoring System. *Proceedings of the 2009 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*. 233-240.

Weiss, D. M., and Lai, C. T. R., 1999. *Software Product-Line Engineering: A Family-Based Software Development Process*. Addison-Wesley.

Yoo, S. E., Kim, J. E., Kim, T., Ahn, S., Sung, J., and Kim, D., 2007. A2S: Automated Agriculture System Based on WSN. *IEEE International Symposium on Consumer Electronics*.