https://doi.org/10.52131/jcsit.2020.0101.0002

*i*RASD Journal of Computer Science and Information Technology

Volume 1, Number 1, 2020, Pages 06 - 25

Journal Homepage:
https://journals.internationalrasd.org/index.php/jcsit

Software Product Line (SPL) Architecture For Context Aware Health Applications

Haris Ali Khan[1], Dr Ali Shahid[2], Dr Amnah Firdous[3]
[1] COMSATS university Islamabad Vehari campus, Pakistan. Email: hariskhan@cuivehari.edu.pk
[2] COMSATS university Islamabad Vehari campus, Pakistan. Email: alishahid@cuivehari.edu.pk
[3] The Government Sadiq College Women University Bahawalpur, Pakistan. Email: amnah@gscwu.edu.pk

ARTICLE INFO

Article History:
Received: July 20, 2020
Revised: October 15, 2020
Accepted: December 20, 2020
Available Online: December 31, 2020

Keywords:
Evolution
Software reuse
Software product line
Feature-based
Software Engineering

OPEN ACCESS

ABSTRACT

Many organizations today are investing in software product-line architecture-for good reason. Software houses are now planning parallel teams to work over similar software but different version with enhanced or modified features. Reuse of unchanged components has not formed the rewards which are undertook because of the components that are reusable are rarely accurate fit to the needs of reuse. Mitigating method of this deficiency is to slight the applicability field to a software product line (SPL). The main problem faced in SPL is to apply the theory and techniques on newer domains, as health applications. Health applications does not require major design or interfacing change, neither a complete revamp in functions is necessary. Health applications is composed of certain features and components which uses particular sensed information, this information comes from interfaced sensors. This research focuses on a software engineering architecture for context-aware health application. The claim of use of SPL is evaluated for the proposed model and architecture. The feature-based SPL model has led to the construction of a mobile application.

Corresponding Author's Email: hariskhan@cuivehari.edu.pk

1. Introduction

A group of systems that are software-intensive share with common and managed features set satisfying the definite needs of a specific market sector or area is known as a software product line (SPL). The key striking SPL part is producing a set of mutual assets, includes planning, analysis, requirement design, test cases, test plans, software reusable components, deployment and other objects (Munir & Shahid, 2010).

SPL engineering has shown an effective and efficient plan to attain economies. Organized variability managing is a fundamental SPL representative. This practice will differ SPL from other software development processes. In SPL we will classify variation points in code and design where to place distinct variants. Management of variation in different phases is very tough. Feature models signify the products in an SPL through dependencies between features and relationships among distinction points. SPL plan handle as, first it is compulsory to check the collaboration of features exposure and Furthermore, it is compulsory to check the accuracy of each product properties. To study for this interest the main two inputs are "a set of product requirements" and the "quality of the variability" model below test. First, the requirement role is to recognize the functionalities of the product and the other is to an emphasis on the test of the end product.

Producing a group of the associated application as an SPL has permitted organizations to attain significant reductions in time and cost and increased quality of product to market. But a product line method to software is assuming both a procedural and a decision based on the business that includes challenges. Cost and advantages of SPL approach for any company, how to manage to produce a product using resources of the product line. How to manage software engineering with SPL engineering? How to structure an organization to organize the production and development of main products and assets and also how to relate software with other new current technologies?

The objective of SPL are the important activities included in different fields of software product lines benefits and cost of accepting a product line method organizational management and technical management performs essentially for achieving effective software product line. Product line engineering patterns help in the adoption of the product line, an implementation roadmap, and analytical method. Product line method can be joint with other technology and as well as business trends, what product line engineering patterns best apply to your organizational way to software product line implementation.

To reduce the development cost and to increase the productivity of software, the software product line engineering (SPLE) is purposed pre-planned the reuse of software artifact at large scale. The SPLE knowledge is to examine the professional field of an applications' family to classify the mutual and flexible parts between all the products. Though, it is mutual for corporations in ad-hoc way to development (e.g. own and clone), a products' group that are going to share common facilities or services and vary in relations of others. So, many current contributions to research proposed reengineering current product to a software product line deviation.

Feature interface is defined usually by means of the behavior of a feature, i.e., interaction is involved in the changing of the behavior of feature, which did not arise when using the features in separation. Interaction between entities may manifest also as attributes of non-functional, in the case when features together have a strong effect on a particular attribute. Also, we have seen feature interaction with respect to the user point of view. E.g. A system that is ready to deliver must be guaranteed by a software engineer that user's intention is full filled by all the features.

Feature-oriented software development (FOSD) to analysis, design implement software, the software development process relies on the concept of feature to which is based on FOSD. The FOSD paradigm is similar to collect tools methods formalism and language connected by feature.

1.1. Context-Aware Applications

With the advancement of technologies now a day's handheld and wearable devices are available to collect the context information to a collection of contextual information is not an issue but the behavior type adoption is still a problem (Martins & Gorschek, 2016). Context-aware adaptation has many goals and the nature of application can determine that goal. Adaptation is built in some applications and some application provides personalized services which are the requirement of users such as location-based app. For the development of context-based applications, many solutions were proposed. But most of them do not have any regular way or systematic approach to develop throughout the development life cycle of application and application that can be enhanced in the future.

Parameters are dynamically adapted by the autonomous systems is expected, behaviors and structures response at runtime to environment and changes in requirements. And these types of systems have four features capability of self-management, i.e. self-optimization, self-configuration, protection of self and self-healing (Salehie & Tahvildari, 2009). Mostly self-adaptation systems have a monitor, analyze, plan and execute (MAPE) (Mizouni, Matar, Al Mahmoud, Alzahmi, & Salah, 2014).

1.2 Health Care and Computer-Mediated Applications

Developed critical security systems in the context of the need for a product line to still obey with protection. Use by the standard the Safety Integrity Levels (SILs) idea to initiate the system's requirements assignment of safety to the under design system component (A. L. de Oliveira et al., 2015). These types of requirements will be used for the health care application. So, we can architect the context-aware health applications according to this scenario. Making common safety integrity levels (SILs) for SPL large scale components by all usage situation considering which is possible, is needed for sale economies

For the safety and critical applications, one of the key requirements of applications is dependability. Most of the applications have changing configurations and context at the time of execution to quality achievement and functional goals and understood as Dynamic Software Product Lines (DSPLs). However, ensuring DSPLs' dependability remains not sufficiently explored, especially in terms of maintainability and reliability. This concessions of quality applicability and assurance of DSPLs in critical safety of areas, e.g. Body Sensor Network (BSN) (Pessoa et al., 2017). In a reasonable cost method of DSPLs is feasible. The benefits of experienced are reliable vigorous monitoring signals for the user health thus providing detection of health issues and the providing the guidance about the health-related information and possibility of proactive treatment from the previously given information.

1.3. The Architecture of the Software Product Line

To develop individualized intensive software systems many sectors of industry face that the demand is increasing and to solve the problem of industrialist engineering in software product line empower the company or organization to build a similar system diversity at minimum cost with high or more quality when it will compare with the development of a single project (Ahmed & Capretz, 2008; Chohan, Bibi, & Motla, 2017). SPLE achievement is the system commonalities belong to the software product line and handles systematically the systems' differences. Commonalities or common features can share properly by all software product line applications or in our context by all software product line context-aware health care applications. E.g. all present feature which is in exiting list of features can be used for any health application development. Inconsistency of Product line defines how the applications changes or vary of the product line can vary. Application of SPL different in the term they fulfill functional, feature and requirements of quality e.g. some cell phones include broadband connectivity of mobile and some are not (Metzger & Pohl, 2014). The SPLE standard record of success is very strong. Some stories of success found in the product line hall of the fame2 book. It has been established as SPLC conference series part 20 listed success stories, including companies Bosch, Boeing, HP, Philips, Nokia, Toshiba, and Siemens (Ahmed & Capretz, 2011; Mariani, Colanzi, & Vergilio, 2016).

The product quality interactions attribute requirements may lead to conflicts of architecture and interfere in the PLAs' assessment. Moreover, evaluation of PLAs is affected by organizational issues, due to their common large number of stakeholder's involved and larger scope.

This research paper is planned as follows: Introduction of this paper depicts the background of software product line information in introduction level and then describe the related work in the term of literature review the work is done on the architecture of software product line, and the context-aware application development process. After the literature review then describes the methodology of research how can be started the thesis or research work. After the design of work will come how the problem statement and research questions can be satisfied and in the implementation phase, a small demo of work is presented.

2. LITERATURE REVIEW

2.1 Software Product Line

Common feature and functionality shared by a set of the similar project known as intensive systems managed by a set of features who satisfied the market and business needs are software product line (Munir & Shahid, 2010).

By using the SPL for health care applications, it can be extended by new features as a feature attribute which will be known as extended SPL. An SPL context-aware health application that will be extended describes the constraints and dependencies among the features of applications through the Extended Feature Model (EFM). An EFM is legally as a hierarchically representation set of functional features arranged and composed by organized relationships as a structure like a tree. An EFM for a health care application Product Line (PL) where a functional feature can be either mandatory or optional, and features can be either qualitative (performance) or quantitative (cost). Moreover, many features can be related to the same feature for, the relations between nonfunctional properties and features. EFMs often encompass cross-tree constraints (CTCs). A new feature adds by CTCs for interdependencies to the feature tree, by checking the selection of connected non-directly optional features. From the requirements, document features are extracted, and add those feature to the FM and this task is not simple to manually perform and could easily a lot of errors to be inserted in the model (Hamza & Walker, 2015).

The software product line will reuse components at a large scale and that is preplanned to get more quality and productivity with low cost (Clements & Northrop, 2002). Software product line (SPL) is based on a group of core assets, that are re-useable software components (Shatnawi, Seriai, & Sahraoui, 2017). SPL is based on two phases application engineering and domain engineering (Schmid & Verlage, 2002). Domain engineering goal is to make core assets that are reusable components based on variability analysis and the commonality of the product family. Software product line

Special architecture is a SPLA that describes the software architecture for a similar project's artifacts (Clements & Northrop, 2002). Previous work is very important for great evolutions. From the source code automatically components based architecture will automatically recover (Shatnawi et al., 2017).

The configuration of different feature and by joining that features without losing any particular specification will derive SPL products (Martins & Gorschek, 2016). SPL may include a huge number of features and they will interact with each other on a large scale and creating complexation of high scale (Martins & Gorschek, 2016). A review of systematic on the domain of SPL approaches of design (Souza Filho et al., 2008).

Domain design methods are studied by them to develop and method applied to activities of SPL, place emphasis on the guidelines, good practices and adopted views by the methods. Furthermore, Murugesupillai, Mohabbati, and Gašević (2011) assumed an initial study of different approaches and their goal are Service-Oriented Architectures (SOAs) and at bridging SPL. And a brief overview provided by them of recent studies approaches classification and type of available research in the area. By the studies of systematic studies, the reason by them is that the studies contributed is the majority (61%) with procedures or methodologies, and the main inspiration factor for SPL-SOA approach was utilizing for the correctly management of variability. The field of quality attributes and SPLs' measures, two studies Montagud and Abrahao (2009) and Montagud, Abrahão, and Insfran (2012) recognized the current lack of appropriate means for many quality attributes measures, and the extensive use of inadequate methods as well for the validation of existing measures. In the single context of systems, Oliveira, Romero Felizardo, Feitosa, and Nakagawa (2010) did a review of systematic focused on situation architectures and models of reference based on reference method that is service-oriented. A panorama is accessible by the author about such model application, importance their uses and report that no consensus at there about how such architectures is to represent. Moreover, reviews of systematic were started in the area of software Architecture Rebuilding by (Ducasse & Pollet, 2009). The activity of evolving architectures is revealed by the studies that have been discussed widely in the last years. Present approaches highpoint utilizing different viewpoints importance and the technical and economic planning need. Though, the best of their knowledge, in the PLA field, no work has been conducted considering the discussed topics in this study. The gap is filled by this work attempt, while underwriting to field structure moreover to other different systematic seven (7) literature reviews attentive on the SPL discipline: Scoping (de Moraes, de Almeida, & Romero, 2009) Requirements, and Testing (Engström & Runeson, 2011; Marques, Simmonds, Rossel, & Bastarrica, 2019).

2.2 SPL for Context-aware Mobile Health Applications

A Software Product Line for the area of mobile and context-aware applications. Mobile are multiuse and multi-sensor apparatus that are supporting applications able to adjust their performance according to variations in the clients' context (device, area, time, etc.) (Lee, Kang, & Lee, 2002). The production of mobile and context-aware application is not an easy task, most of the time the unusual characteristics of these devices. Although some solutions have been planned to ease the development of that type of software, reuse is not systematically used throughout the software development life-cycle. An approach for the development of mobile and context-aware software using the Software Product Line (SPL) paradigm. A Nested SPL for the domain of mobile and context-aware applications is presented (Weiser, 1991). Developed safety critical systems as part of a product line necessity to still obey with safety. Use by the standard the idea of Safety Integrity Levels (SILs) to initiative the system assignment of requirements of safety to the system under design components (A. L. de Oliveira et al., 2015).

2.3. A literature Review of Health Care Applications

The new generation of applications is the wide adoption of mobile communications and ubiquitous computing brought opportunities for companies and government to rethink about healthcare (Solanas et al., 2014). The global world development process represents a difficult challenge and gets attention to cities to gather more people and provide peoples ethical human manners and efficient services. These two tendencies have led to the presence of healthcare application (healthcare mobile applications). The modern concept of the health care application of mobiles is context-aware health applications.

A new field in the combination of public business, health and medical informatics, raising to information delivered and health services or improved through the related technologies and internet. In a wide sense, the characterizes term is a way of thinking, a state-of-mind, an attitude, and an obligation for networked, worldwide thinking, regionally, globally and to improve health care locally through information and communication technology (Eysenbach et al., 2003). Information and communication technology (ICT) is used mostly in the health care applications intercommunication between patients, cares and doctors. ICT will be used in creating SPL architecture for creating a context-aware health application in which all stakeholder will be on the same page of the agreement through information and communication technology.

A literature review of Mobile Applications with the increase of chronic illness globally, it is observed a decrease in availability of health care resources, an increase in the cost of health care result into an immense need to reevaluate the process of health care. Due to these interests, the use of telematics applications and the advanced health care mobile applications are used to reduce the cost (Kao et al., 2018). A lack of main care providers to offer reasonable, good quality healthcare for the growing aging population has created a crucial need to find new different ways to deliver care to frequently ill people. Mobile applications are one of the most reasonable ways to deliver care to the people who are chronically out of health (Kvedar, Coye, & Everett, 2014). Mobile application health widely includes the use of technologies of information and telecommunication, they are combined within wireless and mobile health care applications. The field is including in the broadly use of telecommunication in the mobile network and multimedia technologies in the delivery of health care. The mHealth term was created by Robert Istepanian for the use of "emerging mobile communications and network technologies for healthcare".

2.4. A literature Review of Context-aware

The definition of context is as the natural conditions and situations that either decides application's conduction or in which occasion of an application happens and is intriguing to the utilization (Chen & Kotz, 2000).

In context-aware applications and systems, in deepness, we look at the use of context, context information models and systems that support gathering and distributing context, and applications that accommodate the varying context. Context-aware research clear that it is old but most important in the field of health applications. The difficulties and probable solutions serve as management for scholars hoping to make context-aware computing an authenticity (Chen & Kotz, 2000). Context awareness states to the system ability of both reacting and sensing to changes according to the situation (Yu, Sheng, Younas, & Shakshuki, 2014). Current application of health applications is converting to context-aware application to change according to the situation. Therefore, peoples are used to a different type of context-aware software systems (CASS) (de Sousa Santos et al., 2017).

Used of context can be to describe the condition of an object (place, object or person) that is related to the collaboration between an application and the user, including the application and the users themselves. In mobile services of context-aware, the fast-emerging changes in cellphone technologies are advances to a new type of smart user-centric application on the future pervasive mobile applications on web health (Nalepa & Bobek, 2014; Raento, Oulasvirta, Petit, & Toivonen, 2005; Shen, Peng, & Zhao, 2012).

2.5. SPL Architecture

Software product line architecture is difficult because SPL deals with the process of development that share common architecture for similar projects and controlled changes. The design of programming product offering has all the more great impact on the general execution of the product improvement and at present engaged with the product line business (Ahmed & Capretz, 2008). Building this new concept of healthcare application, the software product line architecture will be involved in the future. Moreover, the study will discuss the main opportunities and key challenges that context-aware health application will provide a common base for more research on the architecture of other applications. The implementation of communication and information technologies within the application of the healthcare segment directed to the concept of electronic health, which is included in increased efficiency and reduced costs (Andrade, 2013).

To improve software Product Line Architecture (PLA) design with the help of styles of architectural through providing its elements for an organizational betterment for about some benefits, as extensibility, maintainability, and flexibility. The design of PLA can be better by using a search based approach of optimization, taking different metrics of account, such as modularization, coupling, and cohesion of feature (Mariani et al., 2016).

To get new other solutions, which consist of PLAs' substitute, the operators of search used by MOA4PLA adjust the organization of PLA by moving and adding its elements. Rules of the architectural style violate to adopt in the PLA, likewise happens what during the evolution of architecture. In experiments put on MOA4PLA (Multi-objective Optimization Approach for PLA Design) (Colanzi, Vergilio, Gimenes, & Oizumi, 2014), got some solutions that the required rules are not satisfied by the covered architectural style, important to a confusing design piece. To satisfy very important architectural style rules throughout the optimization of PLAs, because style adoption makes the system maintenance easier, and fault propagations can reduce due to changes. Future works intend to conduct bigger PLAs' experiments, to evaluate more efficiently the proposed search operators in PLAs with more features and elements. Finally, other works on the future include the operators' creation to aggregate the rules which are associated with other architectural styles' combination (Mariani et al., 2016).

2.5.1. A literature Review of SPL in Context-aware Mobile Application

Now a day's developments of information technologies (IT) applications are leading the beginning of the time of healthcare applications present everywhere at any time. The context-aware health application needs a device in which to get the benefits of the applications, the devices most probably can be mobile phone and wearable systems or devices like watch etc. which have sensors (Kang, Lee, Ko, Kang, & Lee, 2006). Context-aware is to send information from sensors to healthcare application entities as a medium to solve the interoperability issue between sensor developers and the provider of healthcare service. Design of the systematic process of context-aware healthcare applications with the context-aware framework is to show the possibility of the proposed system.

An interaction of features happens when feature performance is influenced to feature by the existence of other features. It is clearly defined that it is not easily the examination of interaction separately of each feature. All the interaction is properly resolved and known, is produced by the strategy and configuration of each effective product and all these are simplified. The behavior of a feature is recognized easily in isolation, resolving and specifying is not a very easy task with other features. Related many issues combination is an efficient approach, e.g. the interactions amount, with the interaction features numbers, human factors, the different types of interaction, and available amount of software requirements and artifacts. For year collaboration of features in a challenging task. The solution for the resolve and identification of interaction are critical. From the last decade strategies of SPL research available literature on the resolution, analysis, and detection of feature interaction. For the development of a single system product, there is an understanding lack of the common artifact, strategies, research gap and actives for SPL interaction (Martins & Gorschek, 2016). For the safety and critical applications, one of the key requirements of applications is dependability. Most of the applications have changing configurations and context at the time of execution to achieve quality and functional goals and realized as Dynamic Software Product Lines (DSPLs). DSPL creates a promising but emerging research area. It includes the following attributes: reliability, availability, integrity, safety, and maintainability. Essential system ability is dependability, software is fixed in a wide variation of critical systems like medical devices, aircraft, and networks, where events occur randomly which run in environments and are thus disposed to system failures. However, ensuring DSPLs' dependability remains not sufficiently explored, especially in terms of maintainability and reliability. This compromises quality applicability and assurance of DSPLs in safety-critical areas, such as Body Sensor Network (BSN) (Blach et al., 2017).

2.6. Feature Implementation

To deal with the evolution of requirements and systematic reuse, RE for SPL will need for SPL products developing. Requirements of SPL need to represent clearly the commonality and information of variability, e.g., through use cases and through feature models (Alves, Niu, Alves, & Valença, 2010). To support the variability information and commonality in SPL, an approach is proposed called Feature-Driven Requirements Engineering (FeDRE). The SPL functional requirements are approached, which specifies in form of use cases according to a model of a feature using well-defined guidelines in a systematic way (R. P. De Oliveira & De Almeida, 2015).

To model the requirement of context at the feature model is important. The feature model of Abu-Matar and Gomaa is extended with new three stereotypes which will administrate how these three groups are measured while applications' members are derived. More, since the focus is to express variability clearly. Representation of this is the minimal requirements of context needed to run a certain feature. For instance, the need for contextual information of relevant place according to latitude and longitude so for this smartphone or Android must be connected via 3G/4G or Wi-Fi. Connectivity with the network becomes necessary in this case for the feature availability. Extension of feature model with <<ContextReq>> stereotype is needed (Mizouni et al., 2014). Members of SPL are derived created on the selection of feature from the model of the feature. Components' derivation of the mobile application members of SPL is twofold.

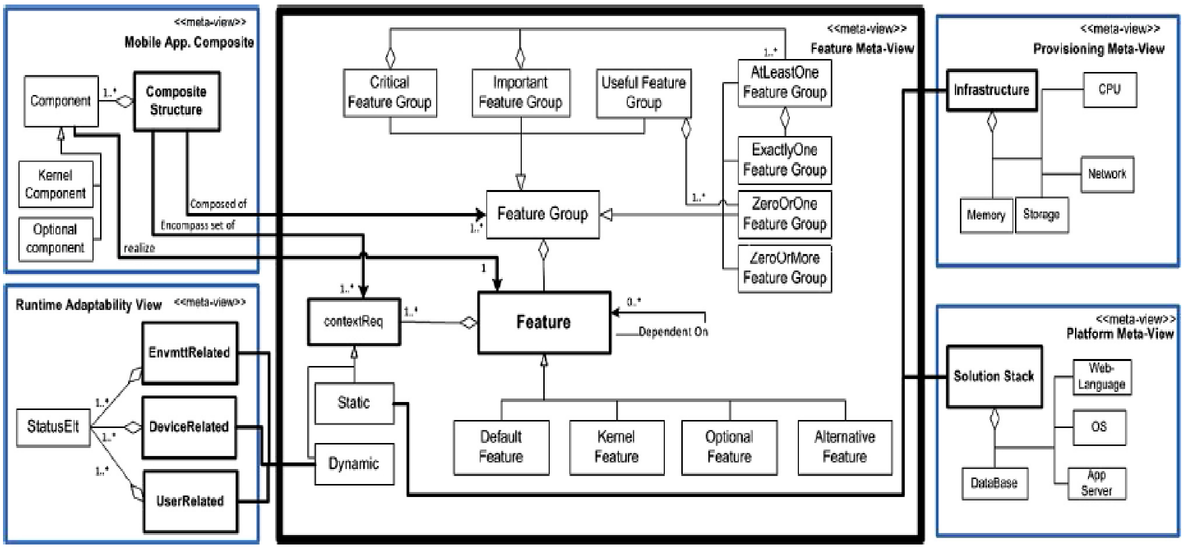

Figure 1: Feature model

The feature is depended on all the groups which are linked to feature. Features in the group of <<Critical>> want to be part of SPL mobile members. Targeting the adaptability (feature availability and feature degradation), Elements which are critically defined by the kernel, and a kernel is a group of compulsory behaviors any SPL member application is needed by the designer to include, and there are some optional behaviors extend by kernel part if the context permits it. Group of <<Exactly-one>> ensures that any member which is generated will, in fact, encapsulate exactly one kernel version from each feature which is critical. In <<Important>> group modeling, Important features are included in this group. Important features are not compulsory to be a part of the kernel but for the user, they are very important. It links to <<zero-or-more>> in the model of the feature. All present feature in the important group is in the manner of derived SPL. The application can change during run time and it has more flexibility of adaptation. During change at runtime important features will be disabled. At design time these types of changes can be handled by the model important feature same as critical features only difference is that it is not a part of the kernel but they are like <<zero-or-more>> group. Display of map and display text need to involve the two other options are similar to <<context>> group, if the present context includes it then it will be derived application part. <<Useful>> group modeling features are measured as optional either the activation will be allowed, current context permits there and they will be made accessible, and these features will be disabled if context does not allow them. Alternatively, they are available. Context requirements are associated with each alternative need to activate that difference. Unlike the other groups, does not kernel is not a part of <<Useful>> features group. Mobile application services do not harm if the kernel part is disabled (Mizouni et al., 2014).

2.7. Feature Model Creation

There are constraints between features (e.g., features are depending on other, one feature requires another feature or one or more features excludes another feature) are domain knowledge which is important, that will correctness guarantee of the configurations of the product in the SPL. The variations used in the SPL extractive process have accurate feature configurations but, separately from these configurations, the combinations of new feature derivation will be desired. Because of feature constraints discovery (mining feature constraints), the step is important. Usually, this is done after the location of the feature because the feature implementation can provide evidence of dependencies of structural between the different features. Though, other approaches might make use of the feature list directly, without the feature location, trying to constraints infer for example feature configurations mining rules of the existing artifacts. Once discovery process of the constraints is finished, structured feature model is created by the feature model synthesis process taken into account the information of constraints and, optionally, other information available from the domain (Martinez, Ziadi, Bissyandé, Klein, & Le Traon, 2017).

Adoption of Extractive SPL is a very difficult task for SPL adopters. On this area needs a common framework compare and test their works on the process of different steps. BUT4Reuse is Extensible and a generic framework that is an open source that aims to challenges facing of these types and that before now has a set of studies of validation.

3. PROPOSED METHODOLOGY

The research goal is research for knowledge in a particular field of study (software product line). To produce or gather knowledge, research is done to increase one's understanding and knowledge of the topic and field or matter in the particular area of research of a person. Research can be defined as a systematic and scientific approach to discovering information that is hidden on a research topic. In their book, Mory and Redman explain "systemized effort to acquire knowledge". A quantitative research model is adopted. The scope given for the SPL research area provides an intention to overview the publications related PLA, our focus at reviewing systematically the literature that resulted in wide coverage, other than focused on narrow analysis. With this work, our purpose is to pay by indicating the quantity of sign in the field, classifying research gaps and trends, and categorizing studies. The method comprises of literature review, software engineering guideline, software engineering and construction, quality analysis and comparison with SPL patterns. Findings of this research will be written as a research paper.

Finally, optimal solutions will be proposed in order to provide ease to organizations as well as the software development community.

3.1. Proposed Model

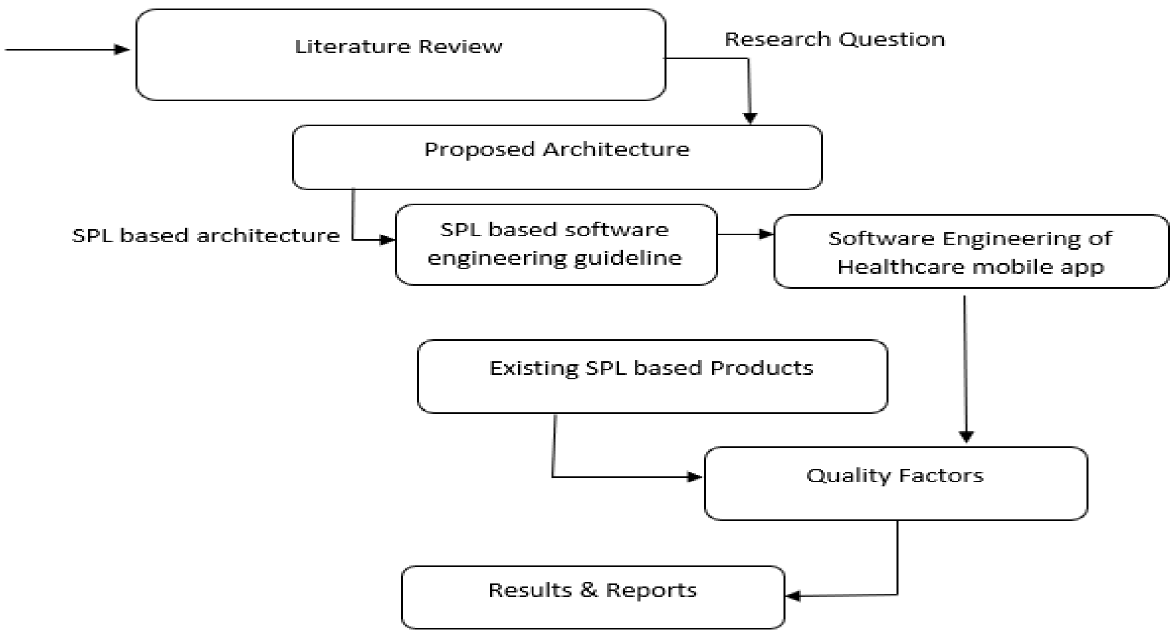

Figure 2: Proposed Model

3.2. Proposed Research Design

A research design is the set of procedure and methods used in analyzing measures and collecting of the variables specified in the research problem. The research design defines the study type (correlation, descriptive, semi-experimental, review, experimental).

3.3. Propose Research Variables Linked with Existing SPL Pattern-based Research will be Adopted

Research projects are mostly based around variables. Variables are straightforward and easily measured such as age, gender or can be a course to be studied. Variable is the characteristic of an individual or a group or an attribute, educational system or research study of interest. Other variables are complicated such as academic achievement, socioeconomic status, or attitude towards school. Environment characteristics are also variables such as the availability of computers of school funding amount. Therefore, the researcher identifies all the variable which are related to his interest after identifying the research topic.

In the proposed model of research methodology first of all literature review is done of different papers for the problem statement that how to select the topic for the thesis. And different research questions are created and these questions are extending after the literature review of more papers. And we get research questions, aim and objective of our thesis on which we work.

According to research questions, SPL architecture for health application is required and for that first of all getting different tools information from different tutorials and research papers. We can search for the development process of different health care application and engineering of the health care application. And then it is compared to the software product guidelines. For designing the model of the feature, we used the Feature IDE software and we built feature model of health care application on that tool and from that model it has created a context-aware health care application development model. And this model is implemented in an android studio and from that, we can get the SPL architecture of context-aware health application and this architecture is given to different software houses for getting feedback e.g.

- Developersarc.com
- Pmis.com
- Zeoz.com etc.

And some other software houses. We get results from there reviews and from different feedbacks. In accordance with the result, most of our thesis objectives are fulfilled. These objectives are categorized in the form of quality factors and those who use the proposed architecture give positive feedback. The feedback of users gives the report and result of the research which is finalized.

4. DESIGN

4.1. SPL Health Care Architecture Assessment

The involving issues of both assessment techniques and architectural refactoring have been widely discussed over the last number of years. In single systems' context, methods aim at assess the consequences of the architectural decision in attributes of quality requirements. The evaluation practice often uses different methods of testing and calculations for checking the complex architecture for developing the product, as the business drivers' conformances are also taken into consideration.

In this thesis work, it is guided by the method presented by (Adachi, Shibasaki, & Kumagai, 2019). The identification representative of four architectural smells discussed by the authors, i.e., design attributes that impact negatively the maintainability of the system. Other than the focusing on artifacts of refactoring implementation, the work speeches the design decisions that are not certainly errant or faulty, but still present an impact negatively in quality of software. The introduced odors were based on the understanding of large two case studies and industrial systems in the literature.

According to knowledge, in the viewpoint of architectural smells in SPLs no studies have been undertaken. Approaches used for dealing with evaluation of PLAs take into consideration the software design description in three viewpoints: functionality, which states to the system features; structure contains an interfaces and components collection; and allocation, makes clear the way in which features are implemented.

The evaluation methods are not interested specifically in architectural smells. Instead, they discuss anti-patterns of software explained by Brown in 1998, in software architecture we argue that they discriminate from smells of architecture in the level of abstraction. While design problems are related by the architecture smells, source code evaluation makes the code smell. So that stakeholders of all type are able to orchestration assess products and features infrastructure in the design level very soon, instead of as the starting point code artifact.

4.2. Feature Model

A set of features that we get from the existing feature which are already built or developed can be extended through SPL by developing the new features according to new requirements with respect to the needs of users. For that extended features SPL describe the constraints and dependencies among the features and nonfunctional properties. An interaction of features occurs when the behavior of feature influenced by the presence of other features. It is defined not easily the analysis of interaction separately of each feature. All the interaction is properly resolved and known, is generated by the strategy and configuration of each valid product and all these are clarified. When the SPL context-aware health application starts growing due to these features can be increased the difficulty to resolve and identify all the interaction. The total interaction of feature candidates is exponential in features numbers, and it is known as the feature interaction problem.

The foremost activity of the phase of feature development analysis (FDA) is feature modeling, which has the objective is to recognize commonalities and variabilities in a domain that is given.

The SPL architecture is designed in the phase of feature design development by formal and informal specifications. Features are developed in feature development integration to meet the specifications. An automatic software group based on tools which support a legal features selection is encouraged by FOSD. There are many academic and commercial tools present to support software engineer's invalid selection finding and product development support, such as Feature IDE for context-aware health application by using the SPL. It is possible to add new features of health application according to the new requirement of health it can be added to the application or it can be dropped with any current exiting feature. SPL for context-aware health application will describe the dependencies among the features and their constraints. Feature models (FMs) for context-aware healthcare application are in a hierarchical manner representation where a type-based feature is assigned on the area of the domain and delivered the context of the ultimate product.

The source code was considered separately of each product, the first step of this work consisted in using the Feature IDE tool and designing a Feature Model and providing each feature the level of abstraction properly for health care application. The features of the context-aware health care application in the feature model are grouped with respect to users' point of view, representing menus dropdown in the GUI (i.e. Format, File, Help, Edit, and Menu). All features represented as leaves of tree state to functionalities that are selectable when originating a product.

The nodes that are inside (except the root) represent a high abstraction level responsible for organizing and grouping the features as the perspective of the user. Circles that are solid connecting to their parents represented as mandatory features, and circles which are hollow, are represented as an optional feature. The Feature Model serves as guidance to develop multiple SPL products. Rules are defined from it for the feature composition and permit stakeholders to acknowledge the product line scope through an overview. SPL Products that are derived must be supported feature model rules.

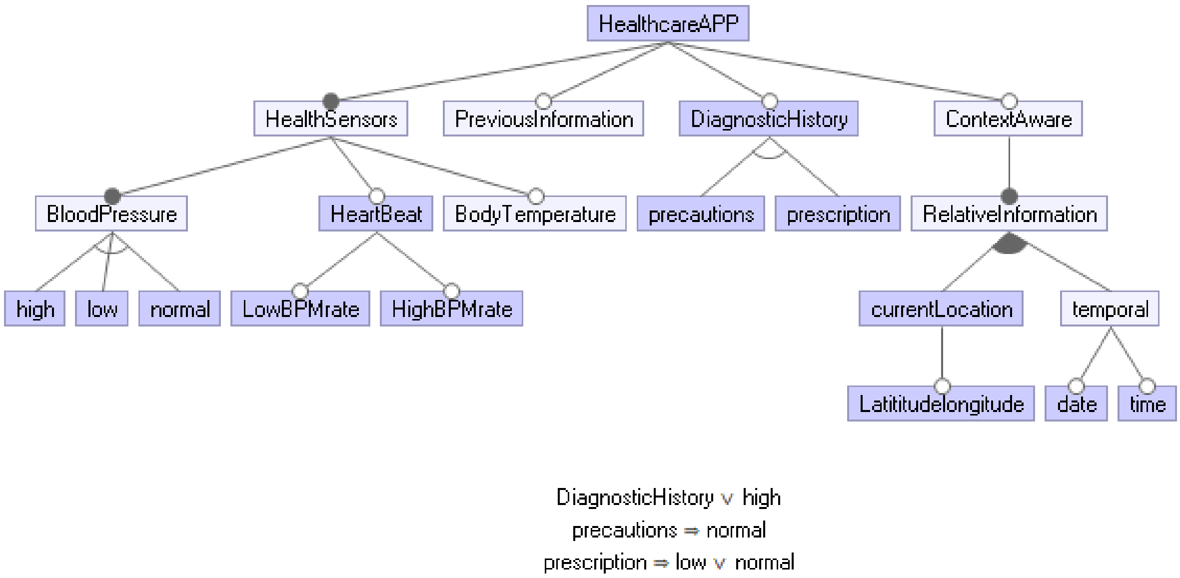

Figure 3: Feature Model

SPL depends on two processes application engineering (AE) and domain engineering (DE). To develop the different artifact later used in the application engineering phase domain engineering is used. For context-aware health application in SPL, domain engineering is used to develop the artifact or feature for a similar product and put it on the existing features and as per requirement, it can be used. And application engineering is to reuse those artifacts which are developed in domain engineering. For the feature, model SPL consists of features which present the characteristics of the final product at the end. The process used for extracting and identifying the features from the software requirements document and add that features to the SPL is error-prone and complex when it is manually done without the tool assistance. In a feature model, features could be mandatory (compulsory in the final product), optional (could be not included or included), XORed (either of the XORed features could be included but not both) or ORed (ORed features each one could be not included or included). By the combination of these are most of the FMs, according to features' selection and their dependencies on one another. Features which are not selected or disabled for the project are shaded but present in the directory of the project. From the documents of requirements features are extracted, and add those features to the FM (feature model) and this is not a simple task to manually perform and could a lot of errors to be inserted in the model easily.

One of the objectives of this thesis is to support the extraction of features from requirements documents and the types and relationships among them. This model is built in Feature IDE, and we implement a feature model in java and using workspace of featureIDE in the eclipse tool. For the context-aware health application qualitative investigations conducted for its validity.

By using SPL for Health care application, we can easily add the features into the application and produce new application of health with new features or changed features without developing the whole new application.

4.3. Health Care Model

To manage the changes in the context-aware health application techniques of the software product line used in this model for the context-aware health application. SPL use common features the for development of a similar type of application so here SPL concept is used to develop the healthcare context-based application by using the common features, new features are also built and saved them into existing features for the development of context-aware health application. SPL goal is a rapid and agile development of the context-aware Healthcare application by using the existing features and new as well throughout the development life cycle.

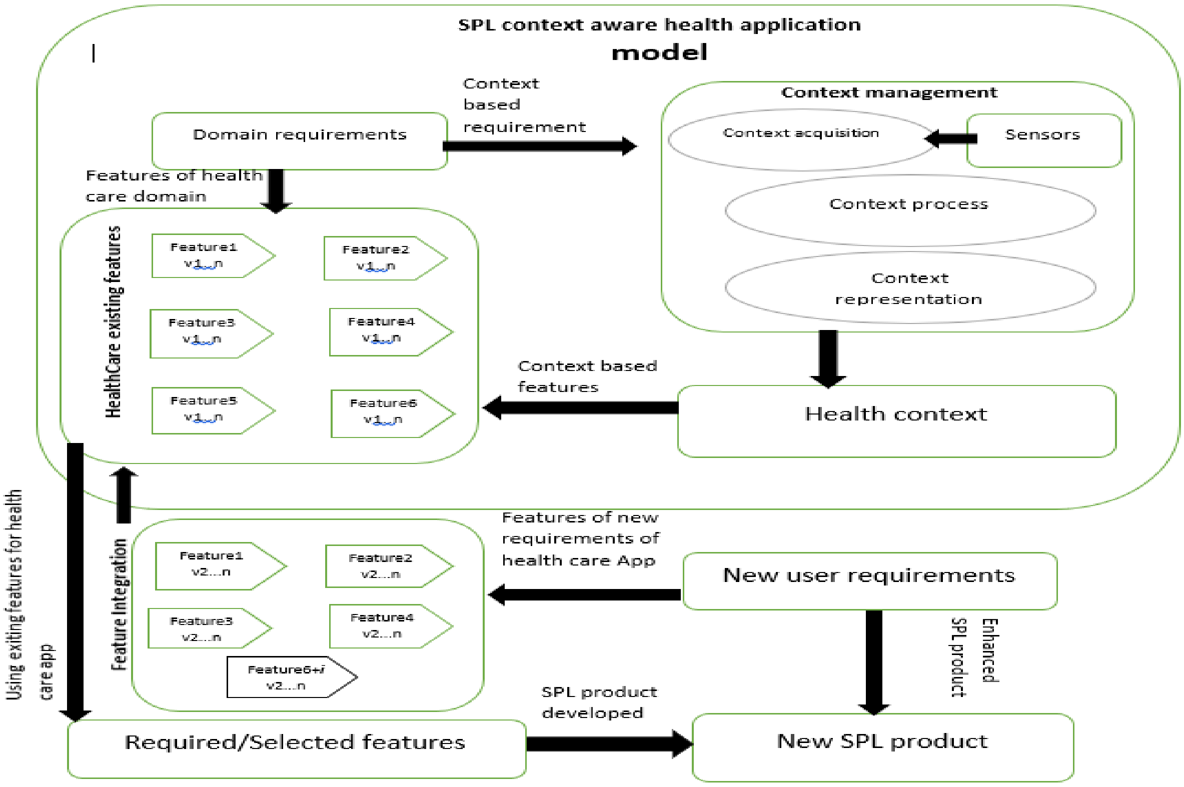

Figure 3: SPL Healthcare Application Model

All the stakeholder will be satisfied through the SPL development model. And through this, we can get the answer to question. Configuration options are provided by the SPLs for the software adjustment for the stakeholders' need and for the different platforms (Pereira, 2017).

In this model we have exiting feature Feature1, Feature2, Feature3, Feature4, Feature5 and Feature6 with respective versions v1 in the beginning. For the development of health care application, we can use features from those exiting features and according to the requirement, we can also create new features v6+i for the development of new product. And that feature can be updated and feature e.g. feature1 version v2, feature2 version v2 etc. So, the new product of health application will be in the product line. And all the needs of stakeholder will be fulfilled by the product. And changes can be accommodated through the SPL architecture and model.

4.4. SPL Architecture for Context-aware Health Applications

SPL architecture for the health care application is created for the development of health care application. For this architecture, we first develop the feature model and then health care application model of SPL and after all that come to make the SPL architecture of the health care application.

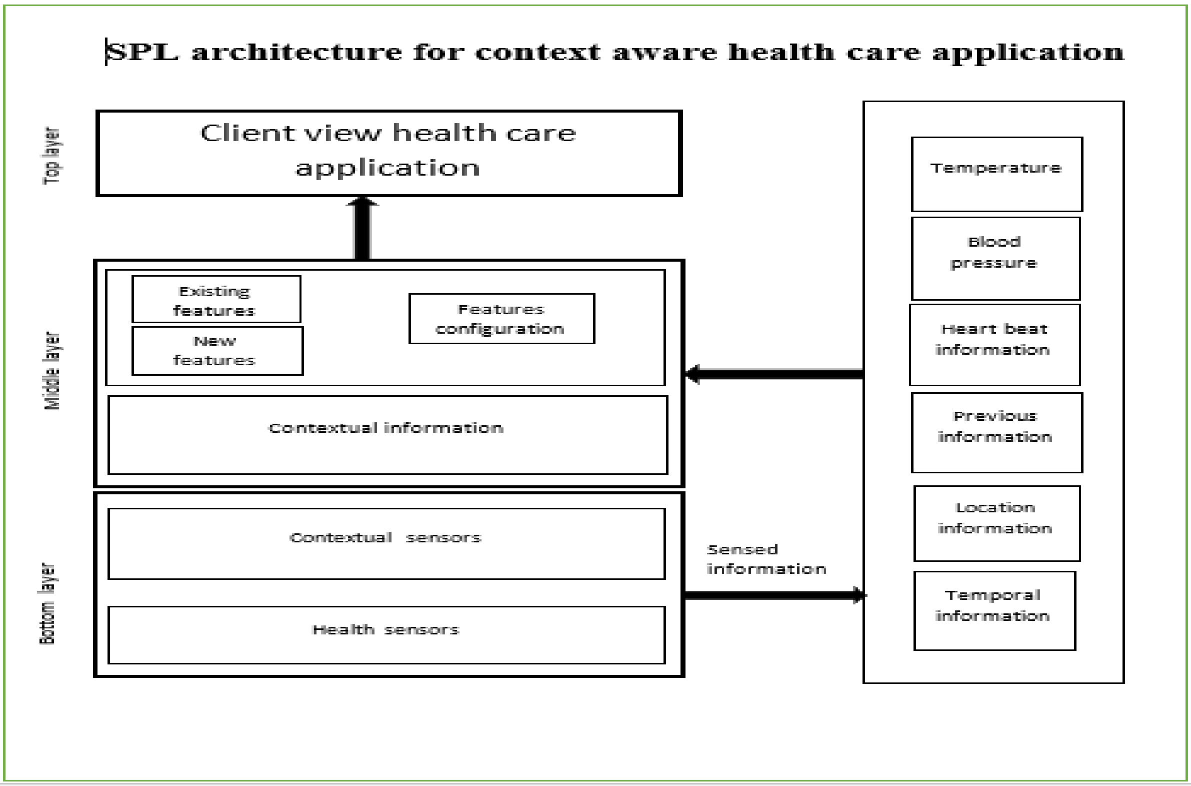

Figure 4: Architecture of Context-Aware Healthcare Application

This architecture will answer the research question four. In this architecture we have sensors in mobile device that will sense contextual information and health information for example, for contextual information sensor will give the information about the latitude and longitude, temporal information (day time etc.) and the health sensor will be sensed information from the body of user and give information about heartbeat, temperate of body and blood pressure etc. and by getting all this information a product will be developed with the existing features and new features can be added according to need and the configuration of the feature will create a new product and user will interact with the product through user interface in the top layer. There is a need to develop a standard architecture for healthcare software product line. The motivation comes from the absence of standard software product line architecture specific for the context-aware health applications. Software product line model for health care applications. This model is developed for the domain of health care applications. This model is created by using the concept of a software product line. By using this model, we can use the exiting features which are already built for a similar project. And according to the new requirement, new features can be created and can be enhanced for a product and its quality is increased by adding new features and those features will be added in exiting feature which will be used in new product whenever will be needed. We can change the feature with a new version of feature and replace old feature with a new version of feature and can get enhanced product same like that we can add a new extra feature that also can be built a product with new functionalities.

SPL engineering has shown to be an effective and efficient plan to achieve economies. Organized variability managing is a fundamental SPL representative. Through a feature model given in figure 4-2, variability can be handled by changing the features without disturbing the other features which are not linked with the variable feature. Product line variability is preplanned like commonalities. It is defined that the given feature is a quality requirement or functional requirement is variability of a product line or not needs a clear decision from a stakeholder or product management. There are two main inputs to consider for this interest: "a set of product requirements" and the "quality of the variability" model under test. First, the role of the requirement to recognize the functionalities of the product and the other is to emphasize on the test of the end product. The new feature can be created to handle the variability. First, the developer can see that is the feature present in the existing repository of features? if yes then the developer will go to use the existing feature and save time and cost of development and if there is no such type of feature is present then he will create a new feature and also will place that feature to the existing repository of the feature. And by the method, we have traceability of product and we can easily manage the changes.

Once was performed the feature location; the features traceability and implementation of the features has been calculated. Reusable assets construction at this point, the goals to extract each feature implementation and prepare it to be used by an SPL annotative or compositional approach. An initial operative SPL has been made possible by the feature model and the reusable assets.

SPLA of context-aware health care application is a special architecture that describes the software architecture for the similar projects of health-related where the features are of similar projects of health. Previous work is very important for great evolutions. Strategies of traceability have been adopted for SPL engineering. From the feature model and health care model it is clearly proved or seen that an application starting from one feature can be enhanced or can come with great evolution after developing and adding new features according to customer needs. And with the help of a software product line, it may not be complex and difficult.

The evolution concept is inherent to software, from when the needs and requirements change over time, so software evolution is needed to remain useful. But, the process of software evolution is quite stimulating since a fragile balance necessity to be maintained, quality of software must be well-maintained but over time the structure of software tends to degrade. Identified challenges are these in the case of SPL evolution. There are different assets types, which are defined at different levels of variability and abstraction, interdependencies between assets are high in numbers. SPL life span is large as compared to single product development and SPL is more complex and larger than a single product. For the health care application using SPL architecture and model, we can have the traceability of application and using the SPL feature model of health care application.

5. Implementation

This architecture and model are used for the development of a simple application of health care. According to architecture, at the start, sensors are not used but only the SPL objective is used for the development. And the applications' features are developed in feature IDE and those features are implemented in the android studio with different classes. This is the main activity

5.1. Results

The architecture is implemented by myself and this architecture is given to people who are related to computer science subject, developers and different software houses as well after given the architecture and model of SPL health care application and take the response from them with a questionnaire
Here is the questionnaire which is filled by the different people groups

6. Questionnaire (Assessment of SPL Architecture)

This questionnaire is conducted for assessing the effect of proposed SPL architecture

1. **Have you used the SPL (Software Product Line)?**
    - Yes
    - No
    - Maybe
2. **Have you used proposed SPL architecture and model for designing your software?**
    - Yes
    - No
    - Maybe
3. **How much time improvement is observed?**
    - 0%-10%
    - 10% -20%

- 20%-30%
- more than 30%
- no improvement

4. **How much improvement is observed in term of ease?**
    - 0 -5 %
    - 5 to 10 %
    - 10 to 15 %
    - No Improvement
5. **How much improvement is observed in term of satisfaction for stakeholder?**
    - 0 to 5 %
    - 5 to 10 %
    - 10 to 15 %
    - No Improvement
6. **What are the effects of SPL architecture on the budget?**
    - Positive
    - Negative
    - No effect
    - Other:

This questionnaire is filled by the 35 persons and they have their own point of views. In these questions all possible options are present. And they filled this form by their own assessment

The output of this survey of each question is here.

Q1) Have you used SPL (Software Product Line)?

Answer of this question in yes is 57% and 34% people does not use SPL previously and 9% answer maybe they use or not used previously.

Response graph

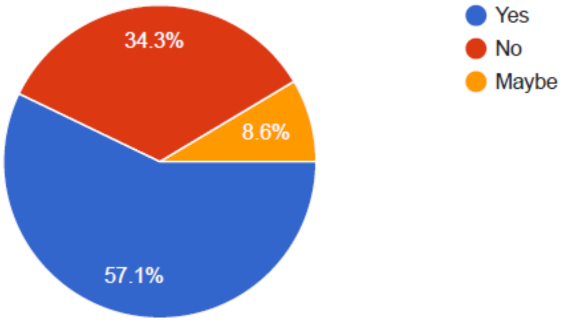

Figure 6: Results of people used SPL

Q2) Have you used proposed SPL architecture and model for designing your software?

Answer of this question in yes is 54.3% and 34% people does not use SPL architecture.

Response graph

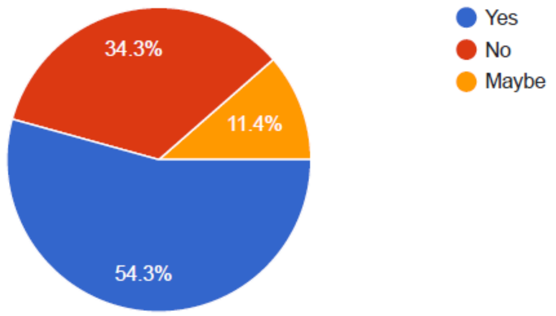

Figure 7: Results of people used SPL

Q3) how much time improvement is observed?

After using the SPL architecture approx. 15% people does not feel any improvement in time and **85%** observe improvement in development time.

Response graph

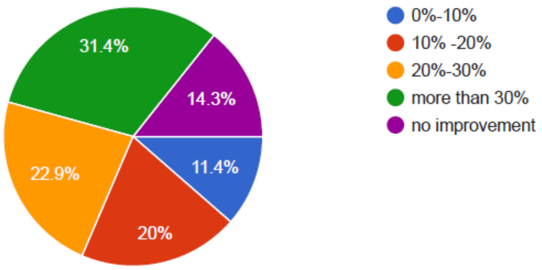

Figure 8: Results of time improvement in development

Q4) How much improvement is observed in term of ease?

After using the SPL architecture approx. 14% people does not feel any improvement in ease and **86%** observe ease in development of new product.

Response graph

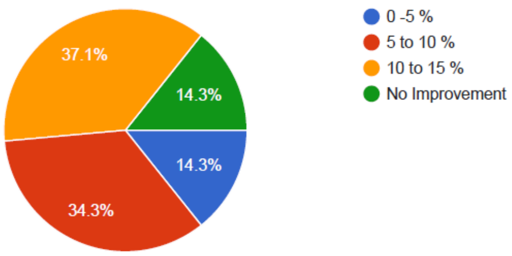

Figure 9: Results of improvement in term of ease

Q5) How much improvement is observed in term of satisfaction for stakeholder?

After using the SPL architecture approx. 88% stakeholders' satisfaction is improved. And 12% stakeholders do not feel any improvement does not feel any improvement in satisfaction.

Response graph

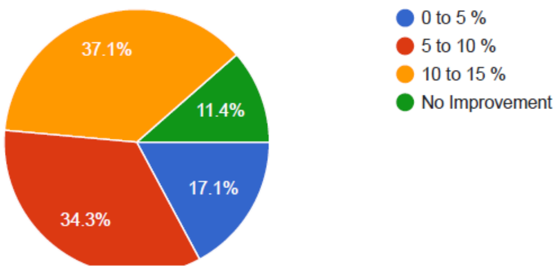

Figure 10: Results of satisfaction level of Stakeholders

Q6) What are the effects of SPL architecture on a budget?

After using the SPL architecture approx. 68% people feel positive effect in budget and 28.6% people observe no improvement and only 3 % people feel negative effect on budget.

Response Graph

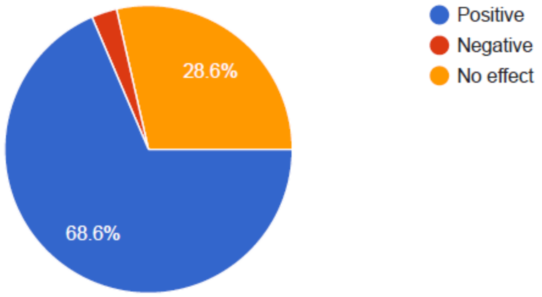

Figure 11: Results of the SPL effect of budget
Results of questionnaire 1

7. Cost comparison of healthcare application

Cost of healthcare application is developed without using SPL and by using SPL with two revisions. First revision of health care application has approximately same cost by using SPL and without using SPL because we developed new features at the start and in second revision development cost decreases and we estimate the cost through cocomo model. By using the reusable component or feature (already developed features) KLOC and feature points decrease and these will decrease more when the number of revisions increases.

App/COST	Function Point	KLOC
Health APP without using SPL	77.04	4.083
Health APP using SPL	77.76	4.121
Health APP without using SPL Revision 2	107.1	5.6763
Health APP with using SPL Revision 2	100.98	5.351

Cost comparison 1

Here in revision 2 of health application functional points are 107.1 without using SPL and with using SPL functional points are 100.98.

8. Conclusion and future work

In this research, SPL is studying for implementation and improvement perspective. We have discovered the latest approaches where feature-based models are being used for application engineering. Also, a software engineering architecture for context-aware health application is proposed using SPL concepts.

The claim of use of SPL is evaluated for the proposed model and architecture. The feature-based SPL model has led to the construction of a mobile application. The application is then developed in Feature IDE for design level. This design is then coded using the android studio.

The outcome of the research is a model and architecture which can lead to design and development of any software engineering application related to a particular domain (here health application).

The engineered model/architecture is given to software engineers for testing and evaluation and a survey is conducted to measure the efficiency and improvement using the proposed approach.

The results of the survey say that 51% peoples used already SPL and 54% used the given architecture and 31% people feel improvement more than 30% and 22 % people feel improvement 20 to 30%. And 31% of people feel improvement of about 0 to 20%. Only 11.4% of people do not feel any improvement. According to the response of people above 80%, people feel ease using the given architecture and only about 10% of people do not feel any ease. And the satisfaction of stakeholders observed by the user of this architecture is also above 80% and 11.4% people are those who do not feel any improvement in this case. And the effect on a budget is felt by the developers is about 68% and 28% of people feel no effect and only 4% of people feel a negative effect.

In future SPL can be used for generic engineering of component-based applications. SPL can be applied to develop interfaced hardware and software. SPL is very suitable for process improvement of software engineering.

References

Adachi, S., Shibasaki, M., & Kumagai, N. (2019). TriQuinoline. *Nature communications, 10*(1), 1-11.

Ahmed, F., & Capretz, L. F. (2008). The software product line architecture: An empirical investigation of key process activities. *Information and software technology, 50*(11), 1098-1113.

Ahmed, F., & Capretz, L. F. (2011). An architecture process maturity model of software product line engineering. *Innovations in Systems and Software Engineering, 7*(3), 191-207.

Alves, V., Niu, N., Alves, C., & Valença, G. (2010). Requirements engineering for software product lines: A systematic literature review. *Information and software technology, 52*(8), 806-820.

Andrade, H. (2013). Software product line architectures: Reviewing the literature and identifying bad smells. In.

Blach, S., Zeuzem, S., Manns, M., Altraif, I., Duberg, A.-S., Muljono, D. H., . . . Negro, F. (2017). Global prevalence and genotype distribution of hepatitis C virus infection in 2015: a modelling study. *The lancet Gastroenterology & hepatology, 2*(3), 161-176.

Chen, G., & Kotz, D. (2000). A survey of context-aware mobile computing research.

Chohan, A. Z., Bibi, A., & Motla, Y. H. (2017). *Optimized software product line architecture and feature modeling in improvement of SPL.* Paper presented at the 2017 International Conference on Frontiers of Information Technology (FIT).

Clements, P., & Northrop, L. (2002). *Software product lines*: Addison-Wesley Boston.

Colanzi, T. E., Vergilio, S. R., Gimenes, I. M., & Oizumi, W. N. (2014). *A search-based approach for software product line design.* Paper presented at the Proceedings of the 18th International Software Product Line Conference-Volume 1.

de Moraes, M. B. S., de Almeida, E. S., & Romero, S. (2009). *A systematic review on software product lines scoping.* Paper presented at the Proceedings of 6th Experimental Software Engineering Latin American Workshop (ESELAW 2009).

de Oliveira, A. L., Papadopoulos, Y., Azevedo, L. S., Parker, D., Braga, R. T., Masiero, P. C., . . . Kelly, T. (2015). Automatic allocation of safety requirements to components of a software product line. *IFAC-PapersOnLine, 48*(21), 1309-1314.

De Oliveira, R. P., & De Almeida, E. S. (2015). *Requirements evolution in software product lines: An empirical study.* Paper presented at the 2015 IX Brazilian Symposium on Components, Architectures and Reuse Software.

de Sousa Santos, I., de Castro Andrade, R. M., Rocha, L. S., Matalonga, S., de Oliveira, K. M., & Travassos, G. H. (2017). Test case design for context-aware applications: Are we there yet? *Information and software technology, 88*, 1-16.

Ducasse, S., & Pollet, D. (2009). Software architecture reconstruction: A process-oriented taxonomy. *IEEE Transactions on Software Engineering, 35*(4), 573-591.

Engström, E., & Runeson, P. (2011). Software product line testing–a systematic mapping study. *Information and software technology, 53*(1), 2-13.

Eysenbach, G., Patt, M., Houston, T., Jenckes, M., Sands, D., Ford, D., . . . Onesimo, R. (2003). Impact Factor (2018): 4.945-ranked# 1 medical informatics journal by Impact Factor Volume 5 (2003), Issue 2 ISSN: 1438-8871 Editor in Chief: Gunther Eysenbach, MD, MPH. *Journal of Medical Internet Research, 5*(2).

Hamza, M., & Walker, R. J. (2015). *Recommending features and feature relationships from requirements documents for software product lines.* Paper presented at the 2015 IEEE/ACM 4th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering.

Kang, D.-O., Lee, H.-J., Ko, E.-J., Kang, K., & Lee, J. (2006). *A wearable context aware system for ubiquitous healthcare.* Paper presented at the 2006 International Conference of the IEEE Engineering in Medicine and Biology Society.

Kao, H.-Y., Wei, C.-W., Yu, M.-C., Liang, T.-Y., Wu, W.-H., & Wu, Y. J. (2018). Integrating a mobile health applications for self-management to enhance Telecare system. *Telematics and Informatics, 35*(4), 815-825.

Kvedar, J., Coye, M. a., & Everett, W. (2014). Early evidence, future promise of connected health connected health: A review of technologies and strategies to improve patient care with telemedicine and telehealth. *Health Aff (Millwood), 33*, 194-199.

Lee, K., Kang, K. C., & Lee, J. (2002). *Concepts and guidelines of feature modeling for product line software engineering.* Paper presented at the International Conference on Software Reuse.

Mariani, T., Colanzi, T. E., & Vergilio, S. R. (2016). Preserving architectural styles in the search based design of software product line architectures. *Journal of Systems and Software, 115*, 157-173.

Marques, M., Simmonds, J., Rossel, P. O., & Bastarrica, M. C. (2019). Software product line evolution: A systematic literature review. *Information and software technology, 105*, 190-208.

Martinez, J., Ziadi, T., Bissyandé, T. F., Klein, J., & Le Traon, Y. (2017). *Bottom-up technologies for reuse: automated extractive adoption of software product lines.* Paper presented at the 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C).

Martins, L. E. G., & Gorschek, T. (2016). Requirements engineering for safety-critical systems: A systematic literature review. *Information and software technology, 75*, 71-89.

Metzger, A., & Pohl, K. (2014). Software product line engineering and variability management: achievements and challenges. In *Future of software engineering proceedings* (pp. 70-84).

Mizouni, R., Matar, M. A., Al Mahmoud, Z., Alzahmi, S., & Salah, A. (2014). A framework for context-aware self-adaptive mobile applications SPL. *Expert Systems with applications, 41*(16), 7549-7564.

Montagud, S., & Abrahao, S. (2009). *Gathering current knowledge about quality evaluation in software product lines.* Paper presented at the Proceedings of the 13th International Software Product Line Conference.

Montagud, S., Abrahão, S., & Insfran, E. (2012). A systematic review of quality attributes and measures for software product lines. *Software Quality Journal, 20*(3), 425-486.

Munir, Q., & Shahid, M. (2010). Software product line: Survey of tools. In.

Murugesupillai, E., Mohabbati, B., & Gašević, D. (2011). *A preliminary mapping study of approaches bridging software product lines and service-oriented architectures.* Paper presented at the Proceedings of the 15th International Software Product Line Conference, Volume 2.

Nalepa, G. J., & Bobek, S. (2014). Rule-based solution for context-aware reasoning on mobile devices. *Computer Science and Information Systems, 11*(1), 171-193.

Oliveira, L. B. R. d., Romero Felizardo, K., Feitosa, D., & Nakagawa, E. Y. (2010). *Reference models and reference architectures based on service-oriented architecture: A systematic review.* Paper presented at the European Conference on Software Architecture.

Pereira, J. A. (2017). *Runtime collaborative-based configuration of software product lines.* Paper presented at the 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C).

Pessoa, L., Fernandes, P., Castro, T., Alves, V., Rodrigues, G. N., & Carvalho, H. (2017). Building reliable and maintainable dynamic software product lines: An investigation in the body sensor network domain. *Information and software technology, 86*, 54-70.

Raento, M., Oulasvirta, A., Petit, R., & Toivonen, H. (2005). ContextPhone: A prototyping platform for context-aware mobile applications. *IEEE pervasive computing, 4*(2), 51-59.

Salehie, M., & Tahvildari, L. (2009). Self-adaptive software: Landscape and research challenges. *ACM transactions on autonomous and adaptive systems (TAAS), 4*(2), 1-42.

Schmid, K., & Verlage, M. (2002). The economic impact of product line adoption and evolution. *IEEE software, 19*(4), 50-57.

Shatnawi, A., Seriai, A.-D., & Sahraoui, H. (2017). Recovering software product line architecture of a family of object-oriented product variants. *Journal of Systems and Software, 131*, 325-346.

Shen, L., Peng, X., & Zhao, W. (2012). *Software product line engineering for developing self-adaptive systems: Towards the domain requirements.* Paper presented at the 2012 IEEE 36th Annual Computer Software and Applications Conference.

Solanas, A., Patsakis, C., Conti, M., Vlachos, I. S., Ramos, V., Falcone, F., . . . Perrea, D. N. (2014). Smart health: A context-aware health paradigm within smart cities. *IEEE Communications Magazine, 52*(8), 74-81.

Souza Filho, E. D. d., Oliveira Cavalcanti, R. d., Neiva, D. F., Oliveira, T. H., Lisboa, L. B., Almeida, E. S. d., & Lemos Meira, S. R. d. (2008). *Evaluating domain design approaches using systematic review.* Paper presented at the European Conference on Software Architecture.

Weiser, M. (1991). The Commputer for the 21st Century. *Scientific American*, 94-104.

Yu, J., Sheng, Q. Z., Younas, M., & Shakshuki, E. (2014). Advances in context-aware mobile services. In (Vol. 18, pp. 1027-1028): Springer.