An Empirical Study of the Improved SPLD Framework using Expert Opinion Technique

Md. Mottahir Alam
(Ph.D. Scholar), Dept. of Computer Science, Singhania University, Jhunjhunu Rajasthan, India

Asif Irshad Khan
Department of Computer Science
FCIT, King Abdulaziz University
Jeddah, Saudi Arabia

Aasim Zafar
Dept. of Computer Science
Aligarh Muslim University
Aligarh, UP, India

***Abstract*—Due to the growing need for high-performance and low-cost software applications and the increasing competitiveness, the industry is under pressure to deliver products with low development cost, reduced delivery time and improved quality. To address these demands, researchers have proposed several development methodologies and frameworks. One of the latest methodologies is software product line (SPL) which utilizes the concepts like reusability and variability to deliver successful products with shorter time-to-market, least development and minimum maintenance cost with a high-quality product. This research paper is a validation of our proposed framework, Improved Software Product Line (ISPL), using Expert Opinion Technique. An extensive survey based on a set of questionnaires on various aspects and sub-processes of the ISPLD Framework was carried. Analysis of the empirical data concludes that ISPL shows significant improvements on several aspects of the contemporary SPL frameworks.**

Keywords- Software Product Line; empirical study; software reuse; feature coverage; variability; product line methods; feature modeling.

I. Introduction

The constant demand for developing larger and complex software applications in a limited time and optimal cost tries to find an answer in software reusability. According to K Jordan [1], software reusability is the approach in which software products or applications are realized by implementing pre-existing software assets in an organization. Assets include specifications, library, design, framework, tools, software program, component or some documentation. There are numerous benefits of reusability [2, 3]:

1) Improve timeliness by delivering product in lesser time to market;
2) Lesser maintenance cost;
3) Boost development efficiency
4) Improve consistency of the software design;

Traditionally, software reuse, in general, was not planned and was focused on maintaining software components into a repository in the anticipation that opportunities for future reuse may happen. Nowadays, a number of optimized software reuse approaches have been proposed by the researchers and industries [3]. One such approach is software product line engineering (SPLE) which supports reuse by building tailored products meeting needs of particular market segments. Reuse in SPL is planned in a systematic and efficient manner and is performed for each artifact resulting from the development process. These artifacts are organized and interrelated collection of software components that can be reused across applications.

SPLE manipulates commonalities and variabilities of a set of similar products in an organized way so as to lower development costs, improve quality, and minimize time-to-market. Under this approach, software products are developed around a set of some common components with specific variabilities resulting in different product configurations [4, 5]. As per practical experiences, SPL helps companies to achieve out of bound improvements in development cost, delivery time, product quality, and adaptability [4].

SPLE is generally comprised of 2 main engineering stages: domain engineering and application engineering [6,7]. During domain engineering, the commonality and the variability in the product line are described and implemented. In application engineering, the different product applications are delivered by reusing artifacts developed during domain engineering and manipulating the product line variability. Product development through SPLE is achieved by developing a flexible and reusable product platform [7]. The basic idea of this developmental approach is to separate common parts of a product family from the differences between the products on the basis of domain knowledge. These common features are used to build a platform which is then used as a standard to create a wide range of products in a product family. In the software context, a platform is a set of reusable assets. It acts as a technological base on which software products or applications are built. The differences among these similar products are captured through variability. Variability is the capability of software artifacts to vary and its smart management and controlling helps in the realization of successful SPL [8]. It is the flexibility of software artifacts to vary and its smart management and controlling help in the realization of successful SPL [9].

Manufacturing companies like Boeing and Ford have already successfully adopted product line approach to have control over budget and efficiency by taking benefit of common features among similar products. Several other companies like Hewlett-Packard, Motorola, Dell have also been following SPL approach [10]. Although product line is not a new idea for manufacturing companies, it is a comparatively new approach for software companies. However, studies have proved beyond doubt that companies which have implemented product line approach have earned tremendous benefits and achieved momentous improvements in costs, delivery time, reliability, usability, portability and maintenance, and customer satisfaction [6, 11].

Bosch in 2000 proposed that creation of a product line can be categorized into three parts, namely, the vital reusable assets of the product line, the management of the product line and the life cycles of the reusable artifacts of the product line [12]. Researchers, of late, have proposed several approaches to implementing software reuse by amalgamating component-based software development (CBSD) and software product lines (SPL) as it maximizes the software reusability.

This paper is planned as follows: Section II discusses related work in software product line framework. Section III discusses the overview of the improved SPL framework. Section IV describes assessment using expert opinion technique. Section V presents the research findings and section VI provides concluding remarks and future work.

II. RELATED WORK

There are several approaches proposed by researchers to manage software product line development.

Weiss et.al [13] put forward a model for SPLE, called as FAST (Family-oriented abstraction, specification, and translation). It is a feature-based model.

FAST Model supports a major share of common artifacts among themselves when the range of similar products is developed.

Common artifacts are analyzed to create a group of products and that result in building of potential software families with common feature like behavior, interfaces, or code. This helps in making the software development more robust by reusing the common artifacts, which in turn decreases the development cost, and reduces the time-to-market as shown in Figure 1.

The processes in this framework are divided into three sub-processes:

i. **Domain qualification**: In this phase cost benefit analysis is done to study the economic feasibility of the potential product line.

ii. **Domain engineering**: In this phase product line infrastructure, core assets reusability and family definition are analyzed for the potential product line.

iii. **Application Engineering**: In this phase reusable core assets are used to build the product line family.

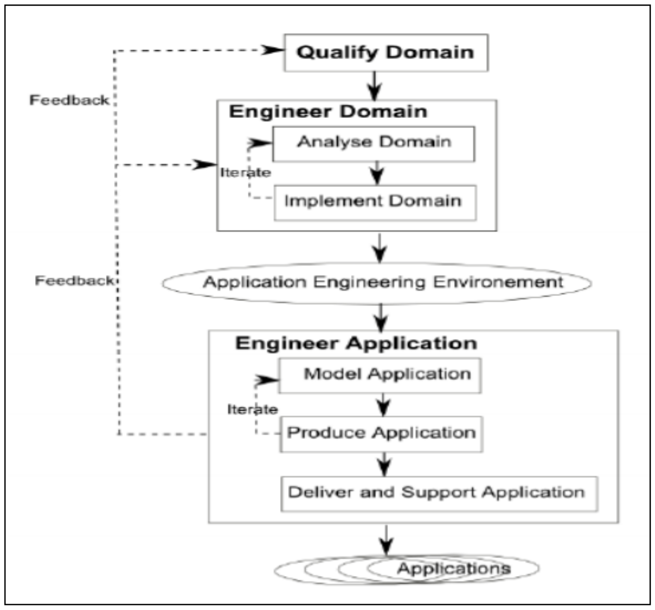


Figure 1. FAST Flow Process [13].

Kang proposed FODA-Feature-Oriented Domain Analysis [14] to identify and model features. This model is based on domain analysis process to identify distinct features within a product line. This approach involves three basic processes, namely,

i) Analysis of the domain of the SPL
ii) Analysis the features of the SPL
iii) Modeling the features of the SPL

Kim, J. et.al [15] in 2008 suggested a framework for implementing both domain requirements as well as modeling core architecture in SPL. It is presented in Fig. 2. The framework uses processes, methods and support tools and shows a mapping between product line requirements and reference architecture. It involves concepts such as goal oriented domain requirement analysis, analytical hierarchy process (AHP), matrix technique and architecture styles.

It performs domain requirement analysis by classifying requirements into four different stages: Business stage, service stage, interaction stage, and internal stage. This helps in identifying and building components.

The next step is to prioritize the components using matrix techniques and analytical hierarchy process (AHP). Finally, reference architecture is created based on the components and their quality attributes.

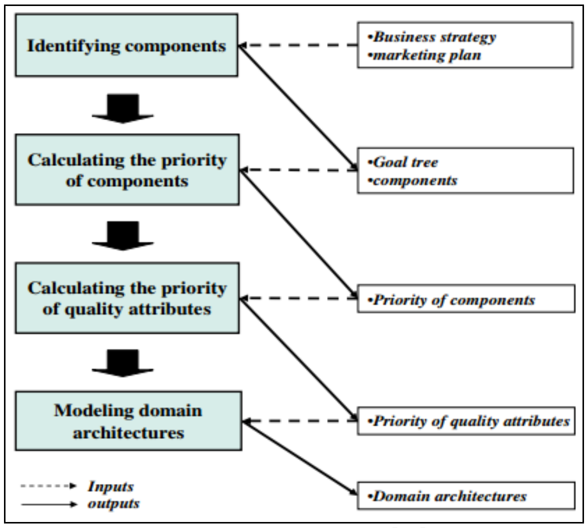

Identifying components
Calculating the priority of components
Calculating the priority of quality attributes
Modeling domain architectures
•Business strategy
•marketing plan
•Goal tree
•components
•Priority of components
•Priority of quality attributes
•Domain architectures
Inputs
outputs

Figure 1. DRAMA process [15].

Tanhaei, M. et.al [16] in 2010 proposed an architecture-based technique to select suitable components in an SPL. It is a component-oriented technique to manage and control the selection of components in an SPL, thereby reducing risks and cost of software development. The components are carefully selected on the basis of the reference architecture, product family requirements, domain requirements, and concerns of stakeholders. The architecture of this method is shown in Fig.3. It starts with the selection of a component list from the component lists on the basis of architecture variant point.

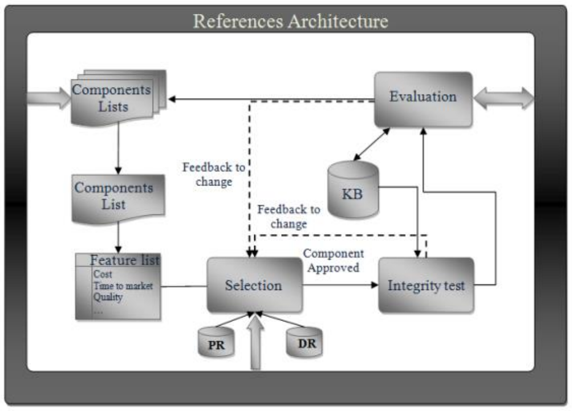


Figure 3. Architecture of method [16].

The components in the component list can be selected either from COTS components or component repository. If the component is not available, then it is developed. The selected components are evaluated for approval. Once approved, these components are passed through integrity test. Lastly, reference architecture is obtained these selected and successfully tested components. Mellado, D et.al [17] gave a framework which emphasizes on security mechanisms for SPL as shown in Fig.4. This framework categorizes activities into two main types: application engineering and domain engineering. It implements security by integrating domain security mechanism PLSecDomReq and application security mechanism PLSecAppReq.

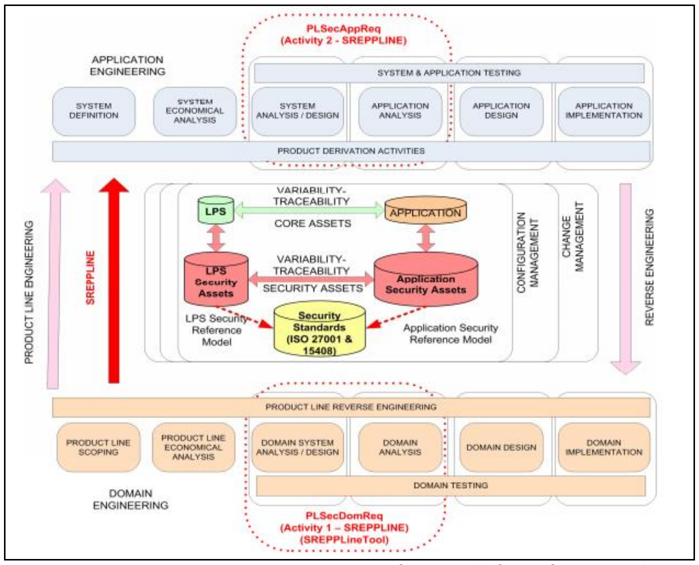

PLSecAppReq (Activity 2 - SREPPLINE)
APPLICATION ENGINEERING
SYSTEM & APPLICATION TESTING
SYSTEM DEFINITION
SYSTEM ECONOMICAL ANALYSIS
SYSTEM ANALYSIS / DESIGN
APPLICATION ANALYSIS
APPLICATION DESIGN
APPLICATION IMPLEMENTATION
PRODUCT DERIVATION ACTIVITIES
PRODUCT LINE ENGINEERING
SREPPLINE
VARIABILITY-TRACEABILITY
LPS
APPLICATION
CORE ASSETS
LPS Security Assets
VARIABILITY-TRACEABILITY
SECURITY ASSETS
Application Security Assets
LPS Security Reference Model
Security Standards (ISO 27001 & 15408)
Application Security Reference Model
CONFIGURATION MANAGEMENT
CHANGE MANAGEMENT
REVERSE ENGINEERING
PRODUCT LINE REVERSE ENGINEERING
PRODUCT LINE SCOPING
PRODUCT LINE ECONOMICAL ANALYSIS
DOMAIN SYSTEM ANALYSIS / DESIGN
DOMAIN ANALYSIS
DOMAIN DESIGN
DOMAIN IMPLEMENTATION
DOMAIN TESTING
DOMAIN ENGINEERING
PLSecDomReq (Activity 1 – SREPPLINE) (SREPPLineTool)

Figure 4. Security requirements engineering framework for Software Product Lines [17].

The reference architecture involves repositories, traceability, and security mechanism. The various repositories implemented in the framework are:

1) Software product line repository
2) Application repository
3) SPLSecurity Assets Repository
4) Application Security Asset Repository
5) The Security Standards Repository

III. Overview Of The Improved SPL Framework

A security enabled framework for software product line development is proposed with a high abstract level of software product line (SPL) architecture as shown in Fig5. The model is a mix of aspect-oriented and the feature-oriented approach. The aspect-oriented approach addresses crosscutting concerns and functional behaviors of SPL while the feature-oriented approach is used to capture variability and commonality of product lines. The detailed explanation of the proposed model is as follows.

The model has two high-level processes: domain engineering and application engineering. The main aim of domain engineering is to identify and develop reusable artifacts for reuse later in the application engineering phase. Application engineering targets building of software products using the identified reusable artifacts.

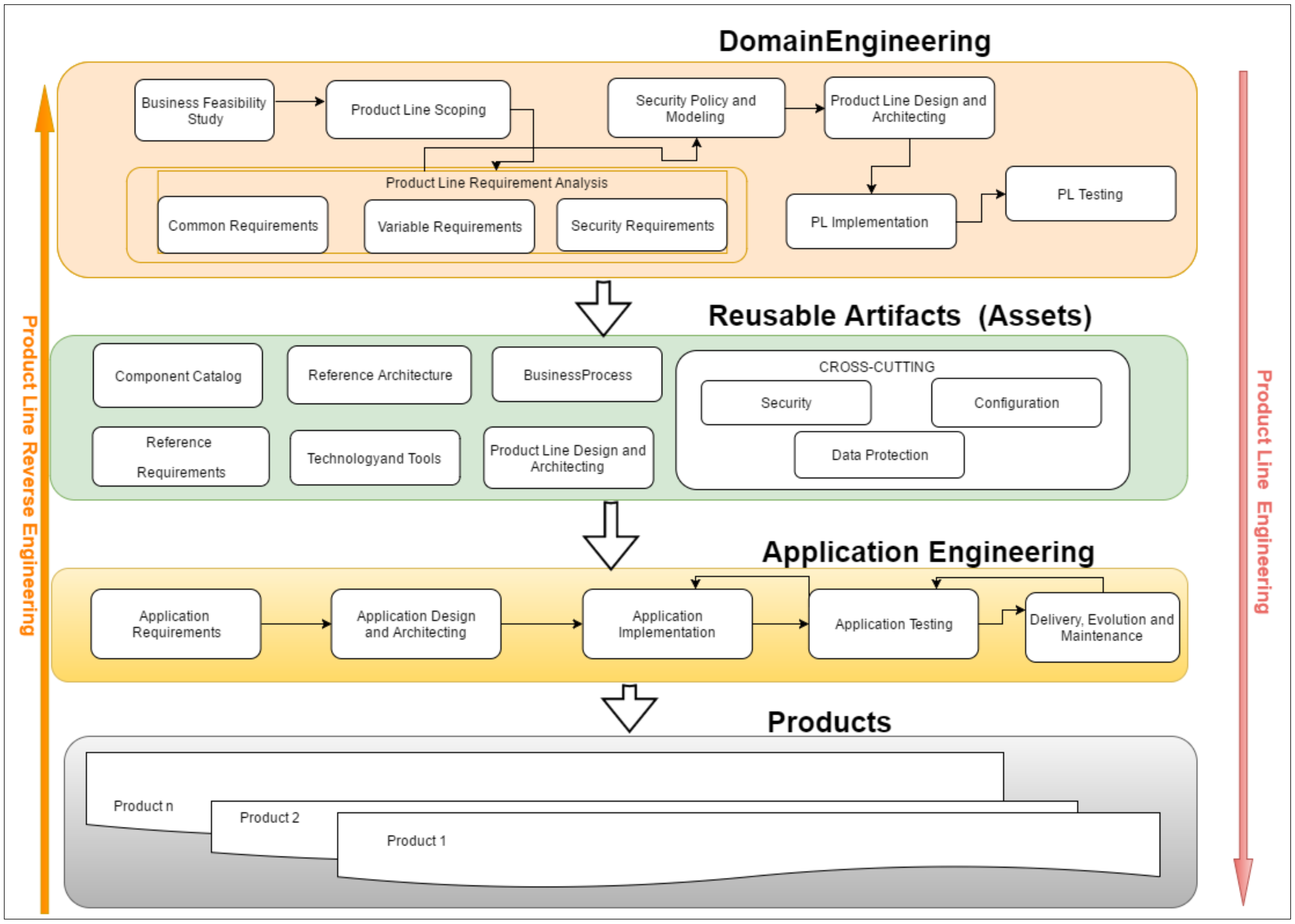


Figure 5. Shows a high abstract level of Software Product Line (ISPL) Architecture.

A. Domain Engineering Phase

Domain Engineering requires common and variable requirements of the product line family as inputs and generates reusable core assets such as components, framework, a library, tools or a platform, etc. The core activities of the domain engineering phase are described as follows:

- Business Feasibility Study
- Product Line Scoping
- Product Line Requirement Analysis
- Security Policy and Security Modeling
- Product line design and architecting
- Product line Implementation
- Product line Testing

B. Application Engineering Phase

Application engineering deals with requirements specifications of individual products of the software product line family are considered, and a customer-specific product is developed by using the generic architecture and reusing the core assets from domain engineering as much as possible.

Following are the major activities carried out to create tailored products during application engineering:

- Application Requirement
- Application Design and Architecting
- Application Implementation
- Application Testing
- Delivery, Evolution, and Maintenance

For a detailed description, refer to [7].

IV. Assessment Using Expert Opinion

Expert opinion technique was utilized to assess and validate different phases of the proposed SPL framework. This method is very useful in evaluating certain system specific questions linked to system behavior, usability and reusability as well as system performance and uncertainties [18]. It also utilizes the wide range of knowledge and expertise of the experts to evaluate the product 19]. Many researchers have used this technique and some scientific publications have emphasized the reliability of using this tool. Authors have conducted expert opinion surveys to validate their system process improvement frameworks and have found this method very trustworthy [20]. Expert opinion technique to solicit the feedback of experts on a process model is has been found to be very useful to validate the framework informally [21].

In this empirical research, the view was sought from experts to evaluate different phases of proposed SPL framework architecture based on the core aims of our research.

A questionnaire comprising of 10 items based on different processes of the proposed framework was created. Experts from software development companies, software vendors, and academics were invited to participate in the research survey.

The following criteria were considered while inviting the survey participants:

1) Currently working as a software architect or system designer or software programmer with more than six years of experience.
2) Should have expertise and skill in using state of the art SPL framework and tools.
3) Should be willing to work as an unbiased evaluator.
4) Should be willing to commit time and effort to offer a valuable opinion based on his working or research experience.

The format of the questionnaire contains statements based on each phase of the proposed SPL framework and respondents were required to give their opinions about the statements in the form of degree of their agreement/disagreement with the statement. We employed the Likert scale which is invented by psychologist Rensis Likert. It is a psychometric response scale and has ratings 1 to 5. It is also called as rating scale and is used by academicians and researchers in surveys to find participants' degree of agreement/disagreement with a specific statement or group of statements. Table1 summarizes the Likert scale employed in this survey.

TABLE 1

5	Very high / Outstanding / Highly
4	High / Considerable / Moderately
3	Nominal / Average / Nominally
2	Low / Nominal / Poorly
1	Very low / Poor / Irrelevant

The results of our analysis were represented using bar charts and frequency tables depicting the extent of analysis. Selection of software experts to invite for the research survey is the most important activity of the empirical research. The basic goal of this empirical investigation and analysis is to validate the proposed SPL framework that ensures a common core SPL architecture achieving write one-time and reuse many- time policy with no or slight variation, which results in reduced development time, improved productivity, and quality product.

V. RESEARCH FINDING

The evaluation of our proposed framework is performed by using a survey method involving software experts from industry as well as academics to validate the different aspects addressed in the given framework. The authors evaluate issues by performing a cumulative evaluation of the various features and concerns in the proposed framework. We sent a questionnaire to 70 software experts in different software companies and colleges around the world. A total of 52 responses was received out of the total 70 questionnaires sent. The remaining 18 couldn't send their responses until the time of our analysis. Of the 52 responses received, six were not included in our analysis as either they were incomplete or didn't meet our criteria for selection. Table 2- shows different abbreviations used in our Analysis Tables and its description.

A. Evaluation of addressing Reusability in the product line

The software product line engineering is based on the concept of software reusability. The proposed framework improves the software reusability in every aspect of the product family. Analysis of the survey concludes that 20% of the responded strongly agreed and 62% agreed that the reusability is addressed in a planned and systematic manner in the framework. Also, approximately 11% of the respondents are unsure of the reusability concern as addressed in the framework while 7% of the respondents don't agree on the reusability concerns in the framework.

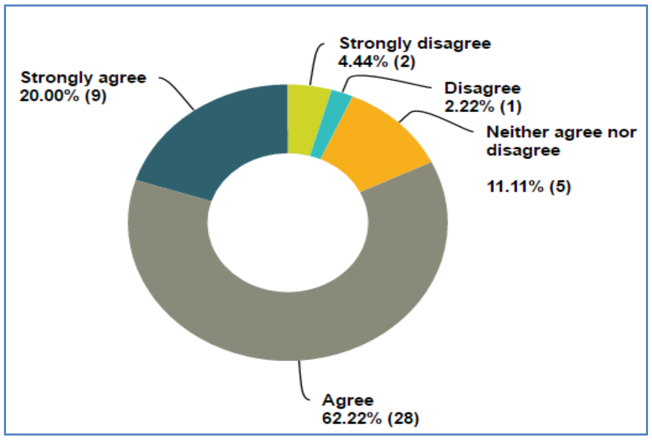
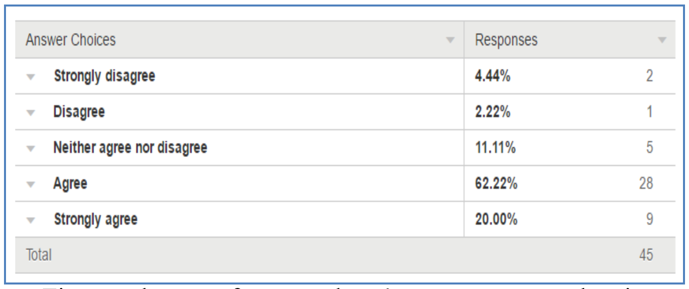


Figure Donut chart of respondent responds to the issue addressed in the proposed framework. Overall, the majority of the respondents gave a positive rating to the proposed framework on this issue.

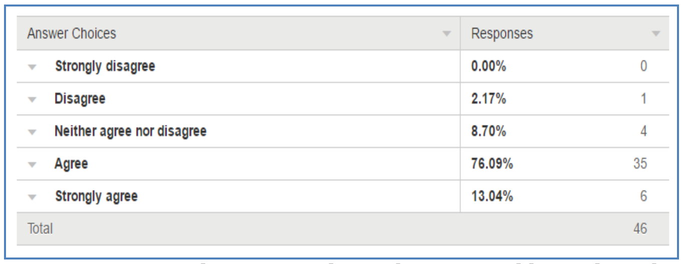


Answer Choices	Responses	
Strongly disagree	4.44%	2
Disagree	2.22%	1
Neither agree nor disagree	11.11%	5
Agree	62.22%	28
Strongly agree	20.00%	9
Total		45

Figure shows of respondents' responses to the issue addressed in the proposed framework.

B. Evaluation of Security, Data Protection, and configuration issues

The framework addressed the cross-cutting concerns such as security and data protection issues in a product line from the early stages of the product line development as retrofitting security in the later stages might break the product line architecture.



Answer Choices	Responses	
Strongly disagree	0.00%	0
Disagree	2.17%	1
Neither agree nor disagree	8.70%	4
Agree	76.09%	35
Strongly agree	13.04%	6
Total		46

Figure respondent responds to the issue addressed in the proposed framework.

As per the survey, 13% strongly agreed while 76% agreed that the proposed framework covers the cross-cutting concerns of the product line in all the stages of domain and application engineering.

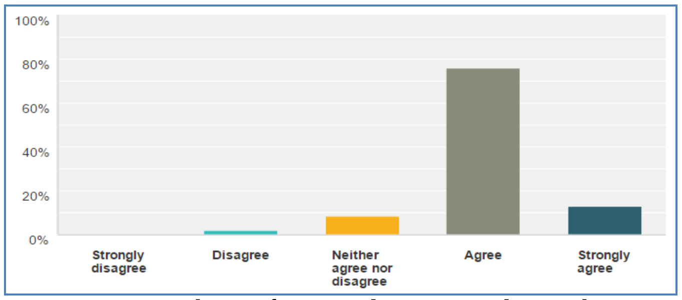


Figure Bar chart of respondent responds to the issue addressed in the proposed framework.

Overall, the framework voted positive rating on this issue. Hence, the respondents considered that our framework succeeded in addressing the cross-cutting concerns from the early stages of the domain engineering till the application engineering of the product family.

C. Evaluation of the commonality and variability aspects

Software product line is built around a set of common software components with points of variability that allow a wide range of products within a product family. We analyzed the responses of the experts on this issue and found that majority of the respondents gave the positive rating to the proposed framework on this issue. As per survey data,

91% of the respondents either strongly agreed or agreed that the proposed framework addresses the key concepts of commonality and variability. Hence, it is concluded that the commonality and variability aspects are also well addressed in the proposed framework.

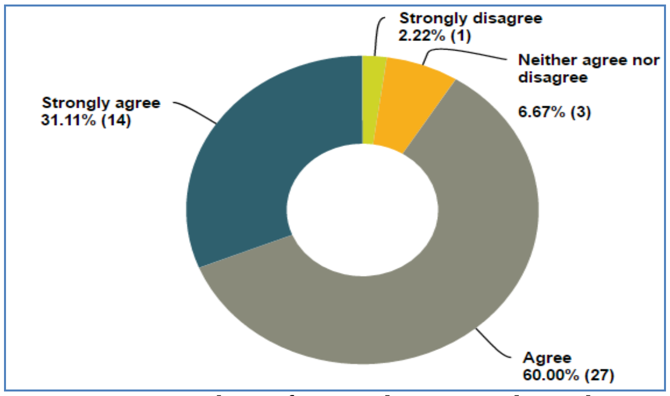
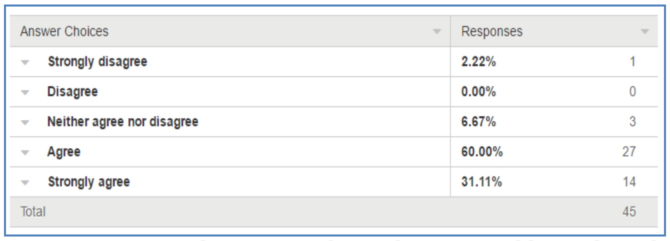


Figure Donut chart of respondent responds to the issue addressed in the proposed framework.



Answer Choices	Responses	
Strongly disagree	2.22%	1
Disagree	0.00%	0
Neither agree nor disagree	6.67%	3
Agree	60.00%	27
Strongly agree	31.11%	14
Total		45

Figure respondent responds to the issue addressed in the proposed framework.

D. Result of evaluation for the use of ADLs to support variability

Architecture description languages (ADLs) provide support for capturing variation points. It is recognized as an important element in the description and analysis of software properties. It allows describing of variable and dynamic features in the SPL. However, not all ADLs provide support for capturing variation points.

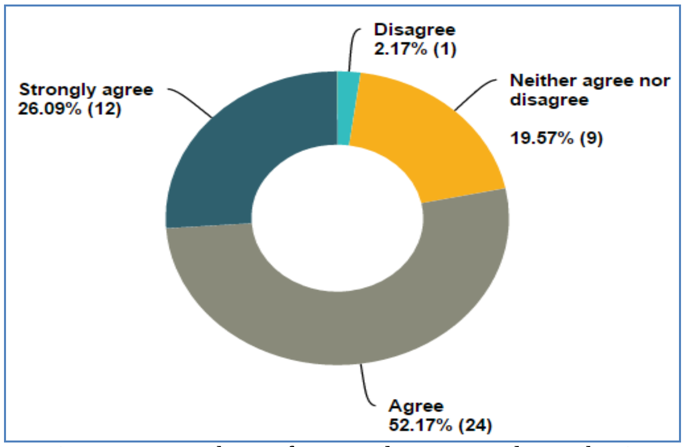
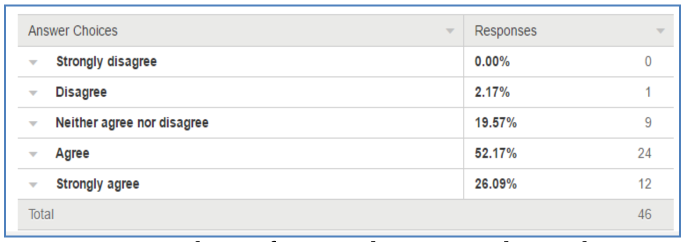


Figure Donut chart of respondent responds to the issue addressed in the proposed framework.



Answer Choices	Responses	
Strongly disagree	0.00%	0
Disagree	2.17%	1
Neither agree nor disagree	19.57%	9
Agree	52.17%	24
Strongly agree	26.09%	12
Total		46

Figure Bar chart of respondent responds to the issue addressed in the proposed framework.

The use of variability-supportive ADLs in the Product line design and architecting phase of the proposed framework is validated, and it is found that 26% of the respondents strongly agreed while 52% agreed that ADLs used in the SPLE should be supportive variability. Overall, 78% respondents agreed that the use of variability-supportive ADLs as suggested in the proposed framework is the best way to incorporate variability in the SPLE. Also, approximately 20% of the respondents were neutral on this matter.

E. Evaluation of Documentation, Tagging and Repository aspects

Documentation is one of the reusable assets under the concept of software reusability in the SPLE. In this proposed framework, various aspects of a software product line development such as business feasibility study, scope, objectives, and evolution of the product line are recommended to be documented. Besides, the product line requirements, architecture design documents, test plan, test design, test cases, etc. should also be developed and well documented. It should also include component's design, its documentation, its test design and procedures, and any models to predict its behavior. All these documents must be maintained so as to be used as reusable assets throughout the lifetime of the product line.

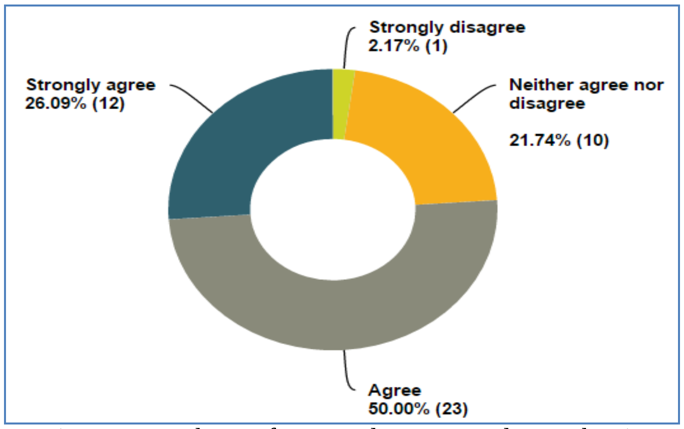
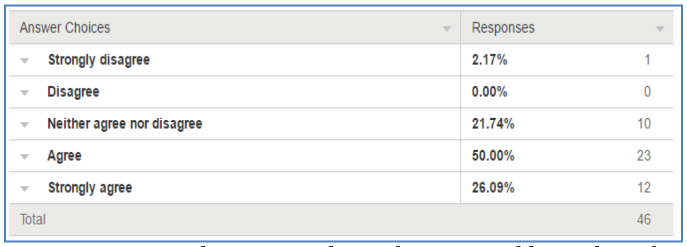


Figure Bar chart of respondent responds to the issue addressed in the proposed framework.

The analysis of the data shows that proposed framework through experts' opinion and found that 26% of the respondents strongly agreed while 50% agreed that reusable artifacts present in the component catalog and repository should be documented and maintained throughout the lifetime of the software product line as mentioned in the proposed SPL framework.



Answer Choices	Responses	
Strongly disagree	2.17%	1
Disagree	0.00%	0
Neither agree nor disagree	21.74%	10
Agree	50.00%	23
Strongly agree	26.09%	12
Total		46

Figure respondent responds to the issue addressed in the proposed framework.

Overall, 76% of the respondents gave a positive rating to the framework.

F. Evaluation of the configuration management aspect in the proposed framework

In a software product line, individual products are built, using selections and configurations of the reusable components as well as the implementation of product-specific features. The configuration comprises of the selection of possible variants for a specific application.

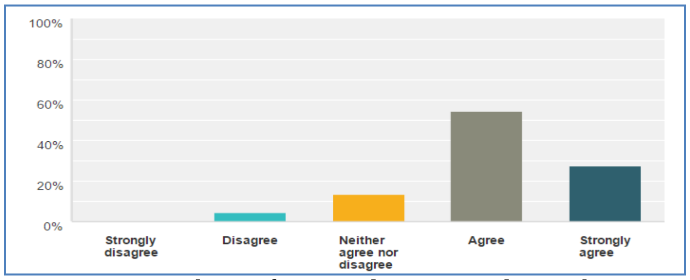
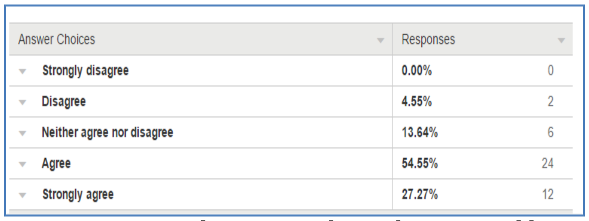


Figure Bar chart of respondent responds to the issue addressed in the proposed framework.

The products are abstracted by configuring the architecture and tailoring components available in the component catalog. It picks and configures the compulsory parts of the reference architecture and adds in product specific variations to yield unique products of a software product family.



Answer Choices	Responses	
Strongly disagree	0.00%	0
Disagree	4.55%	2
Neither agree nor disagree	13.64%	6
Agree	54.55%	24
Strongly agree	27.27%	12

Figure respondent responds to the issue addressed in the proposed framework.

As per the survey data, the majority of the respondents agreed that the configuration aspect is well managed in the proposed framework. 27% of the respondents strongly agreed while 54% agreed that the configuration management is covered in the proposed SPL frame. Also, around 4.5% didn't agree while around 13% didn't have any opinion on this issue. Hence, the empirical study positively validates the proposed framework on this aspect.

G. Result of evaluation to validate whether the proposed framework addresses the testing concerns

In an SPL, every product of a product family is based on a unique set of features. Since SPLE technology is applied to increasingly complex domains, the need for an efficient product testing approach becomes more critical. Furthermore, a product line grows over time to meet new demands of the market, some enhancements or corrective modifications. To make sure that these changes in are in conformance with the product line core architecture, do not bring unforeseen errors, and the new features work as expected, rigorous testing needs to be performed. As per the survey data, the majority of the respondents agreed that the testing aspect is well covered in the proposed framework. 11% of the respondents strongly agreed while 68% agreed that the configuration management is covered in the proposed SPL frame. Also, around 2.2% didn't agree while around 15.5% didn't have any opinion on this issue. Hence, the empirical study positively validates the proposed framework on the testing aspect.

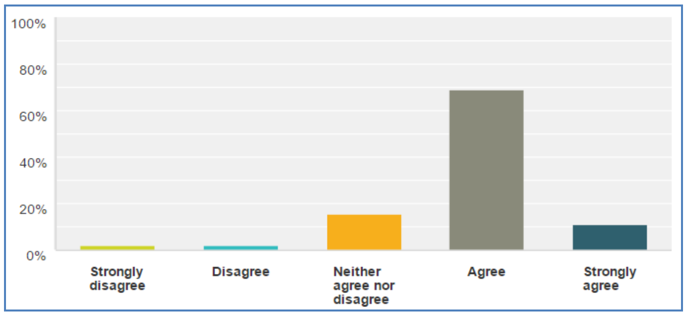
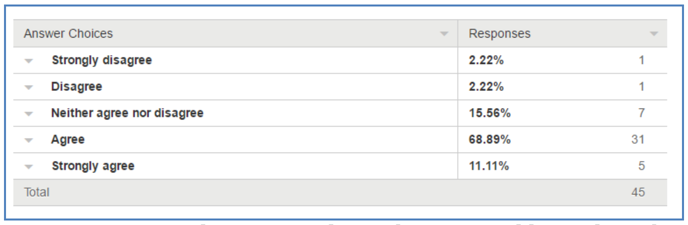


Figure Bar chart of respondent responds to the issue addressed in the proposed framework.



Answer Choices	Responses	
Strongly disagree	2.22%	1
Disagree	2.22%	1
Neither agree nor disagree	15.56%	7
Agree	68.89%	31
Strongly agree	11.11%	5
Total		45

Figure respondent responds to the issue addressed in the proposed framework.

H. Overall Evaluation Results for the ISPL Framework

Overall, most of the evaluation criteria for the ISPL Framework got positive ratings, and hence, it is concluded that the experts found significant improvement in our ISPLD framework.

Clearly, the framework thrives to achieve improvement in almost every phase of the software product line development, i.e. from business feasibility, study and product line scoping to application implementation and testing.

VI. Conclusion

The proposed framework, ISPLD, has been empirically studied and analyzed in this paper using expert opinion technique.

We conducted an extensive survey based on a set of questionnaires on various aspects and sub-processes of the ISPLD Framework. The participants were chosen from software product development firms, software vendors, and academicians. Survey result shows that majority of the expert give positive feedback on the ISPL Framework. The result of analysis concluded that ISPLD framework brings significant improvements.

References

[1] Kimberly Jordan. Software Reuse, Term Paper for the MJY Team Software Risk Management WWW Site, 21 April 1997. Accessed: February 2010, www.baz.com/kjordan/swse625/docs/tp-kj.doc.

[2] Bertrand Meyer. Object-Oriented Software Construction. Prentice Hall PTR, 2nd edition, 2000.

[3] Alam, M.M, Khan, A.I, Zafar, A., "A Comprehensive Study of Software Product Line Frameworks", International Journal of Computer Applications (0975 – 8887) Volume 151 – No.3, October 2016.

[4] Clements, P., Northrop, L.: Software Product Lines: Practices and Patterns. Addison-Wesley (2001)

[5] Van der Linden, F., et al.: Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering. Springer (2007)

[6] K. Pohl, G. B¨ockle, and F. J. v. d. Linden, Software Product Line Engineering: Foundations, Principles, and Techniques.Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[7] Alam, M.M, Khan, A.I, Zafar, A. Md. Mottahir Alam, Asif Irshad Khan, and Aasim Zafar. A Secure Framework for Software Product Line Development. International Journal of Computer Applications 159(4):33-40, February 2017.

[8] M. Svahnberg, J. van Gurp, and J. Bosch. A taxonomy of variability realization techniques. Softw., Pract. Exper., 35(8):705–754, 2005.

[9] Khan, A.I., Alam, M.M., Jedaibi, W.A., January 2015. "Variability Management in Software Development using FeatureIDE: A Case Study." International Journal of Scientific & Engineering Research, Volume 6, Issue 1, 584 ISSN 2229-5518

[10] SPL Hall of Fame, http://splc.net/fame.html

[11] K. Mohan, B. Ramesh, and V. Sugumaran, "Integrating software product line engineering and agile development,"Software, IEEE, vol. 27, no. 3, pp. 48–55, 2010.

[12] Bosch, J. (2000). Design & Use of Software Architectures, Addison-Wesley

[13] Weiss, D., Software Synthesis: The FAST Process, Proc. Computing in High Energy Physics 95, September 1995

[14] K. Kang, S. Cohen, J. Hess, W. Nowak, and S. Peterson. Feature Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie MellonUniversity, Pittsburgh, PA, USA, November 1990.

[15] Kim, J., Park, S., Sugumaran, V.,2008. "DRAMA: A framework for domain requirements analysis and modeling architectures in software product lines." Journal of Systems and Software, 81(1), 37-55.

[16] Tanhaei, M., Moaven, S., Habibi, J., 2010. "Toward an architecture-based method for selecting composer components to make software product line." Proceedings of the 2010 Seventh International Conference on Information Technology: New Generations (ITNG) (pp. 1233-1236).

[17] Mellado, D., Fernández, M. E., Piattini, M., 2010. "Security requirements engineering framework for software product lines." Information and Software Technology, 52(10), 1094-1117.

[18] Ali B M, Kitchenham B, Assessment of a Framework for Comparing Software Architecture Analysis Methods. Proceedings of 11th International Conference on Evaluation and Assessment in Software Engineering, BCS, 2007.

[19] Matthias R, Tools & Techniques Expert Opinion, Online available http://www.idemployee.id.tue.nl/g.w.m.rauterberg/lecturenotes/UFTexpertopinion.pdf

[20] Sajid R., Moving Towards Component Based Software Engineering in Train Control Applications, Final thesis, Linköpings universitet, 2012, Sweden.

[21] Kitchenham B, et al., An empirical study of maintenance and development estimation accuracy. Journal of Systems and Software, 2002, 64 (1), 57-77.

AUTHOR PROFILE

Mr. Md Mottahir Alam is a Ph.D. scholar in the Computer Science & Engineering in Singhania University, India. He has six years of experience as Software Engineer (quality) for leading software multinationals, where he worked on projects for companies like Pearson and Reader's Digest. He is ISTQB Certified Software Tester. He has received his Bachelor's degree in Electronics & Communication and Masters in Nanotechnology from Jamia Millia Islamia University, New Delhi, India. Mr. Alam research interest includes Software Engineering esp. Software Product Line Engineering, Software Reusability, Component-based and Agent-based Software Engineering.

Dr. Asif Irshad Khan is working as a faculty member in the Department of Computer Science, FCIT, King Abdulaziz University, Jeddah, Saudi Arabia. He has thirteen years of experience as a professional academician and researcher. Dr. Khan received Ph.D. in Computer Science and Engineering from Singhania University, Rajasthan, India, and Master & Bachelor degrees in Computer Science from the Aligarh Muslim University (A.M.U), Aligarh, India. He has published several research articles in leading journals and conferences. He is a member of the editorial boards of international journals, and his current research interest includes Software Engineering with a focus on Component Based and Software Product Line Engineering.

Dr. Aasim Zafar is working as an Associate Professor in the Computer Science Department, AMU, Aligarh. His current research interest includes e-learning, mobile learning, virtual learning environments and mobile ad hoc networks. He received his Ph.D. degree in Computer Science from the Aligarh Muslim University, India. He has a number of research papers to his credits. Dr. Zafar is a member of Internet Society (ISOC). He is a member of several editorial boards and a regular reviewer for reputed journals.

© 2017 by the author(s); licensee Empirical Research Press Ltd. United Kingdom. This is an open access article distributed under the terms and conditions of the Creative Commons by Attribution (CC-BY) license. (http://creativecommons.org/licenses/by/4.0/).