Customizing a Feature Ontology for Product Line Engineering within a System-of-Systems Context

2019 International Symposium on Systems Engineering (ISSE)

Tekinerdogan, B.; Duman, S.; Caner, H.; Durak, B.

https://doi.org/10.1109/ISSE46696.2019.8984459

Customizing a Feature Ontology for Product Line Engineering within a System-of-Systems Context

Bedir Tekinerdogan
Information Technology
Wageningen University
Wageningen, The Netherlands
bedir.tekinerdogan@wur.nl

Sami Duman
ASELSAN
Ankara, Turkey
duman@aselsan.com.tr

Hakan Caner
ASELSAN
Ankara, Turkey
hcaner@aselsan.com.tr

Bülent Durak
ASELSAN
Ankara, Turkey
durak@aselsan.com.tr

Abstract— **System of systems (SoS) is an arrangement of systems that results when independent systems are integrated into a larger system that delivers unique capabilities that cannot be provided by the constituent systems. Product line engineering (PLE) is a large scale comprehensive reuse approach to reduce the cost of development, reduce the time-to-market, and increase the quality of the system. In this paper, we report on our experiences in developing a PLE approach for a SoS context, that is SoS-PLE. The results are derived from an action research within the context of Aselsan, a large scale systems engineering company. One of the challenges for SoS-PLE is the selection and customization of the required ontology for commonality and variability modeling for the SoS context. Various different feature modeling approaches have been provided in the literature, but these have largely focused on single PLE. We show the analysis and selection of a feature ontology for SoS-PLE. Subsequently, the selected feature ontology is enhanced and integrated within the product line engineering process. The results of the paper provide novel insight for the state-of-the-practice in PLE for systems engineering, and elaborates on and complement the existing literature on feature ontologies and SoSs.**

Keywords—system of systems engineering, product line engineering, feature modeling, method engineering

I. INTRODUCTION

A system-of-systems (SoS) is an arrangement of systems that results when independent and useful systems are integrated into a larger system that delivers unique capabilities [4][5][6][15]. SoSs have been largely applied in the defense domain but obviously, SoSs are also apparent in many diverse application domains. Developing SoS is not trivial, and cumbersome. SoS development is often considered as a special case of systems engineering. Systems engineering is an interdisciplinary approach to translating users' needs into the definition of a system, its architecture and design through an iterative process that results in an effective operational system. Systems engineering applies over the entire life cycle, from concept development to final disposal. While systems engineering has focused on defining a systematic life cycle process to meet the quality requirements, reuse has largely been an implicit concern.

To reduce the cost of the developing SoSs reuse approaches can be applied. Reuse has been an important goal in many industrial practices and broadly addressed in the literature. While reuse was initially focused on small scale, ad hoc reuse, currently it is widely recognized that the broadest and the most valuable benefits are derived from a large-scale systematic reuse approach. This idea has culminated in the product line engineering (PLE) approach that indeed focuses on exploiting reuse over the whole lifecycle process. Traditionally, a product line is defined as a set of systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way [3][12]. Despite earlier reuse approaches, PLE aims to provide pro-active, pre-planned reuse at a large granularity (domain and product level) to develop applications from a core, shared asset base. With this large scale systematic reuse it is envisioned to reduce the development costs, reduce the time-to-market, and increase the quality of the products. [2][13] While PLE has initially focused on software reuse the idea has further and is now also being applied in the systems engineering domain, leading to the notion of *systems product line engineering*.

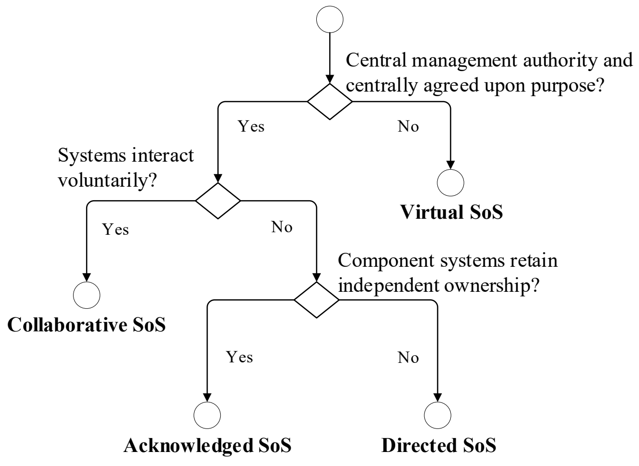
In this paper, we report on our experiences in developing a PLE approach for SoSs, that is SoS-PLE. The results are derived from an action research that has been carried out within the context of Aselsan, a large scale systems engineering company [1]. In this context, one of the challenges for SoS-PLE appeared to be the selection and customization of the required ontology for commonality and variability modeling. Various different feature modeling approaches have been provided in the literature but these appear to have largely focused on single PLE and do not directly align or scale with the larger scope of SoS. We show the results of our analysis on the popular state-of-the-art variability modeling approaches with respect to their applicability for SoS. Based on the analysis a feature ontology is selected for SoS-PLE. We adopt and enhance the selected feature modeling approach and discuss the integration of the ontology within a PLE process for SoS. The results of the paper provide novel insight for the state-of-the-practice in PLE for systems engineering, and elaborates on and complement the existing literature on feature ontologies and SoSs.

The remainder of the paper is organized as follows. In section 2 we present the background. Section 3 presents the adopted research method for developing the integrated systems PLE. Section 4 presents the analysis of the feature modeling approaches. Section 5 presents the selected feature ontology for the integration in SoS-PLE. Section 6 presents the related work, and finally section 7 concludes the paper.

II. Background

A. System-of-Systems Engineering

Obviously, one of the frequently cited classification of SoS is based on the management of the constituent systems. The initial classification from this perspective was provided by Maier [4] who distinguished among directed, collaborative and virtual SoSs.



Central management authority and centrally agreed upon purpose?
Yes
No
Systems interact voluntarily?
Virtual SoS
Yes
No
Component systems retain independent ownership?
Collaborative SoS
Yes
No
Acknowledged SoS
Directed SoS

Fig. 1. Different types of SoS (adopted from: [8]) based on management policy

Directed SoS are built and managed to fulfill specific purposes. The SoS is centrally managed to fulfill the agreed purpose. The component systems can operate independently, but their normal operational mode is subordinated to the central managed purpose. *Collaborative SoSs* have a commonly agreed purpose but do not have a central management. The component systems interact voluntarily to fulfill agreed-upon central purposes. *Virtual SoSs* lack a central management authority and a centrally agreed-upon purpose. Further, the component systems have their independent ownership. *Acknowledged SoS* have also a recognized objective and central management. In contrast to Directed SoS the constituent systems retain their independent ownership. Changes in the systems are based on collaboration between the SoS and the system. Fig. 1 shows a diagram (in BPMN) that shows the different types of SoS from this classification perspective. This paper focuses on PLE within an SoS context of a single organization. Thus, we can state that the approach largely applies to directed SoSs.

B. Product Line Engineering Process

Different product line engineering processes have been proposed in the literature. In general, the PLE process consists of two different activities. In the domain engineering the focus is on developing reusable assets, while in the application engineering these reusable assets are used to develop products. The terms domain engineering is also called *core/reusable asset development*, while application engineering is sometimes termed *product development*.

The common PLE process is shown in Fig. 2. . This is the traditional first generation PLE process in which both the horizontal (within domain engineering, within application engineering), and the vertical transitions (from domain engineering to application engineering) are manual. For example, application requirements engineering is based on a manual process for analyzing existing reusable domain requirements assets.

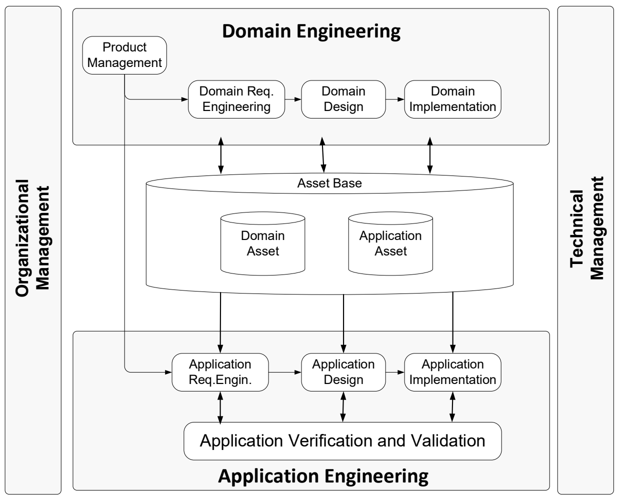


Fig. 2. First Generation SPLE Process

Recently, enhancements have been provided for the PLE process that focus on an automation of the activities. This so-called second generation-PLE embodies a more well-defined and repeatable process, centered on a strong factory paradigm. In this process, a configurator tool takes as input a feature-based description of a product and exercises the variation points (VP) in the shared assets to produce an artifact set that supports the named features. Product development thus becomes automated, so that application engineering which is important in first-generation approaches becomes very small. The overall process for 2G-PLE is shown in Fig. 3. This figure shows the overall PLE process without considering the systems engineering process.

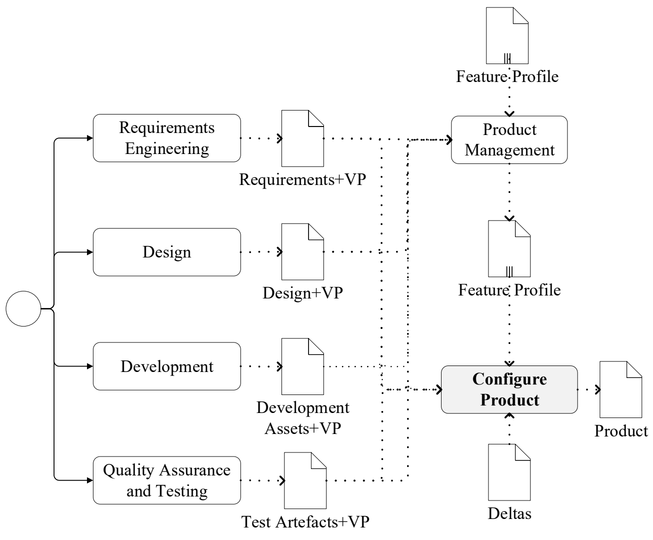


Fig. 3. The 2G-PLE factory paradigm

C. Variability Modeling

Variability Modeling is one of the key activities in the product line engineering context. Over the last decade, several variability modeling techniques have been developed that are aimed to support variability management during product derivation. In this context, *feature models* were introduced in the Feature-Oriented Domain Analysis (FODA) method by Kang in 1990 [7]. Since its introduction feature modeling has been widely adopted by the software product line community and a number of extensions have been proposed**.** A feature is a system property that is relevant to some stakeholder and is used to capture commonalities or discriminate between systems.

A *feature model* is a model that defines features and their dependencies. Feature models are usually represented in feature diagram (or tables). A *feature diagram* is a tree with the root representing a concept (e.g., a software system), and its descendent nodes are features. Relationships between a parent feature and its child features (or subfeatures) are categorized as:

- *Mandatory* – child feature is required.
- *Optional* – child feature is optional.
- Or – at least one of the sub-features must be selected.
- *Alternative* (xor) – one of the sub-features must be selected

A *feature configuration* is a set of features which describes a member of an SPL. A *feature constraint* further restricts the possible selections of features to define configurations. The most common feature constraints are:

- A requires B – The selection of A in a product implies the selection of B.
- A excludes B – A and B cannot be part of the same product.

Besides of the basic variability model as defined by FODA different extensions have been proposed. A nice classification of these approaches is defined by Sinnema and Deelstra [14] who list, among others, the following variability modeling approaches: FeatureRSEB, RequiLine, Cardinality-Based Feature Modeling (CBFM), ConIPF, PureVariants, GEARS, OVM, VSL, and Gomaa. All of these techniques provide a modeling approach for variability to support variability management. They share lots of commonality but are different in terms of modeling concepts and in terms of the tools that support them.

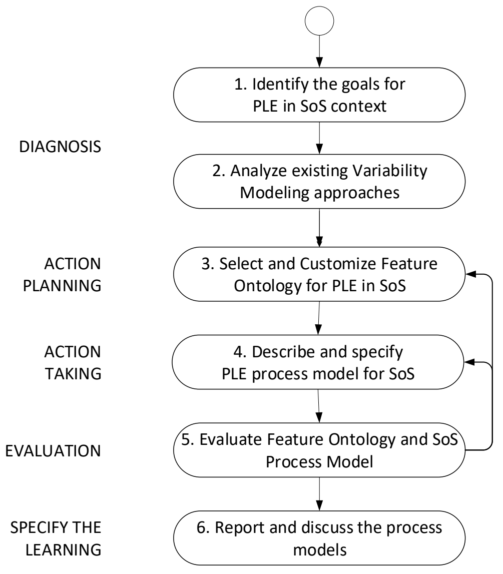
III. Research Method

The overall objective of this study is the identification of a variability model for the system-of-systems context, and develop a method adopting the variability model. The research has been carried out within the context of ASELSAN whereby the main reason is to enhance reuse and likewise reduce cost, reduce time-to-market of the developed products and increase quality. In particular the following research question was defined:

RQ1. What is the required feature ontology for the product line engineering process in a system-of-systems engineering context?

RQ2. How to integrate variability modeling in the PLE process for system-of-system?

The defined research questions require a thorough understanding of the literature and also the development of a novel method. For this, we have adopted an empirical research approach based on so-called action research which appears to be an important and valid instrument for solving research and development problems within an industrial context. The adopted steps in the research are shown in Figure 3.



DIAGNOSIS
1. Identify the goals for PLE in SoS context
2. Analyze existing Variability Modeling approaches
ACTION PLANNING
3. Select and Customize Feature Ontology for PLE in SoS
ACTION TAKING
4. Describe and specify PLE process model for SoS
EVALUATION
5. Evaluate Feature Ontology and SoS Process Model
SPECIFY THE LEARNING
6. Report and discuss the process models

Fig. 4. Overall approach for modeling the SPLE process.

IV. Selecting and Modeling Feature Ontology

To select a suitable variability modeling approach we have first listed the requirements that we think are essential in the context of SoS-PLE. Inspired from the classification framework for classifying variability modeling approaches as defined by Sinnema and Deelstra [14] we analyzed each approach with respect to the expressiveness for modeling features, modeling formal constraints, modeling traceability to artefacts, abstraction mechanisms for coping with complexity, and tool support. In addition to these criteria we deliberately analyzed each approach for its explicit consideration of SoS. From this analysis we selected the so-called feature ontology of Krueger et al. [8][9] which is also the subject of an upcoming ISO standard that is in progress with involvement and support from INCOSE through its Product Line Engineering International Working Group [6]. The so-called enterprise ontology includes the following terms [9]:

Feature Catalog is a model of the collection of all of the feature options and variants that are available across the entire product line.

Bill-of-Features is a specification for a product in the product line portfolio, rendered in terms of the specific features from the Feature Catalog that are included in the product.

Bill-of-Features Portfolio is the collection of Bills-of-Features for the entire product line. Portfolio teams, typically working with product marketing teams, create

Bill-of-Features for product families based on the features available in the Feature Catalog.

Shared Assets are the digital artifacts associated with the systems and software engineering lifecycle of the product line. Shared assets typically include requirements, source code, test cases, user documentation, a bill of materials (parts lists) and more.

PLE Factory Configurator is the mechanism that automatically produces application assets for the digital twin of a specific product. The configurator provides the abstraction-driven automation the eliminates the labor intensive and error-prone activity of manually assembling and modifying engineering assets for the digital twin for each product in the product line.

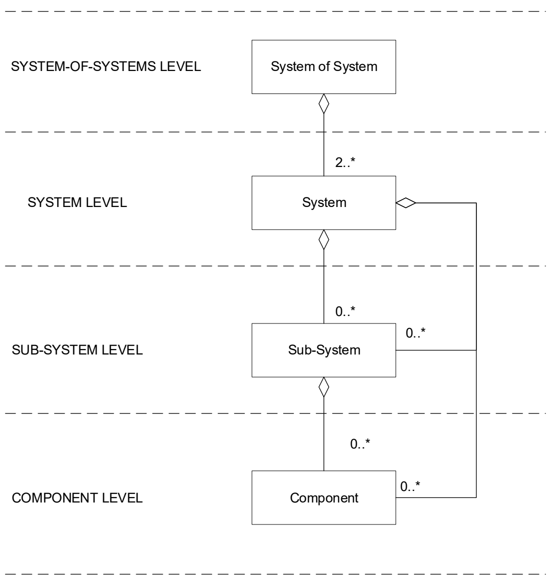


Fig. 5. Metamodel Systems engineering in Feature-Driven PLE

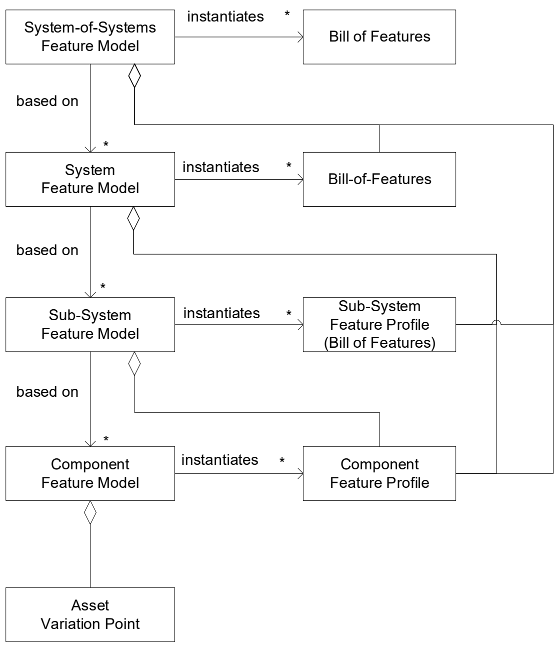


Fig. 6. Metamodel Feature Modeling in Feature-Driven PLE

Based on the analysis of the feature ontology we extracted and modeled the perspective on SoS and the feature modeling approach. The metamodel representing the structure of SoS in the enterprise ontology is defined as in Fig. 5. As we can observe from the figure an SoS consists of systems, which consist of sub-systems, and sub-systems consists of components. A system can also directly include components. The metamodel of the feature ontology of the feature-driven PLE approach is shown in Fig. 6. The bill of features are instantiated from a SoS feature model, which is then gradually refined and selected up to the component level. For more detail about the feature-driven PLE we refer to [8][9].

In the previous sub-sections we have discussed the basic metamodel and the ontology for feature-driven SPLE. In the original studies [8][9] an implicit bottom-up process is defined. We have modeled this as shown in Fig. 7. Hereby, the process starts from the shared assets, from which primitive standalone feature models are developed, component feature models, sub-system feature models and system feature models.

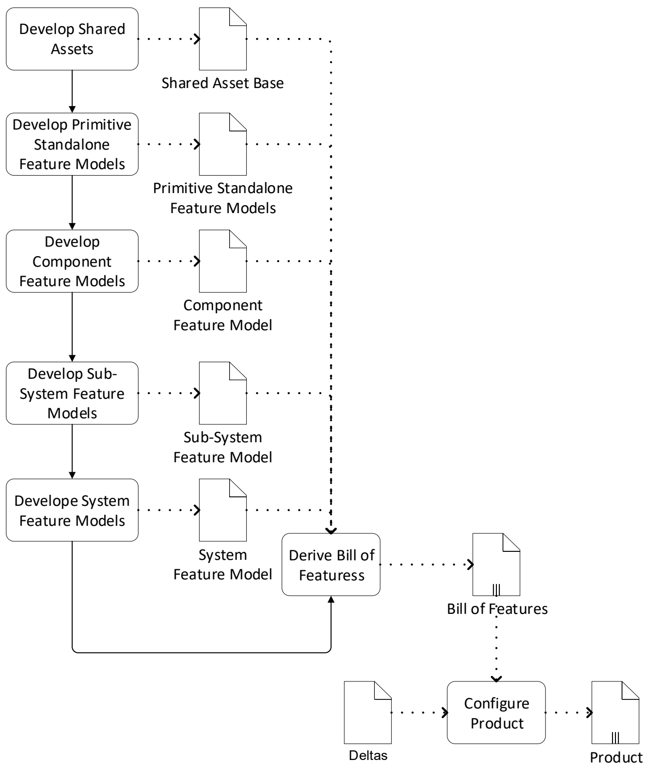


Fig. 7. Bottom-Up PLE Process for Feature-Driven PLE

V. Customizing Feature Ontology

The ASELSAN process is in essence a second generation PLE process that focuses both on systems engineering and automation. In addition, it builds on the feature-driven PLE process as described in the previous section. The ASELSAN Proline process however differs in three ways from the feature driven SPLE approach. First of all, the system metamodel for Proline is more refined than that of feature-driven SPLE. Second, as a consequence of the different system metamodel, the corresponding feature ontology (metamodel) is also customized. Thirdly, the

ASELSAN process adopts (also) a top-down approach rather than a bottom-up approach.

Fig. 8 shows the conceptual view for the adoption, customization and application of the selected feature ontology. The top-level *Feature Ontology* is selected from the literature. The *Customized Feature Ontology* is based on this ontology but is customized for the company and SoS context. Finally, the *Applied Feature Ontology* defines the used Customized Ontology in a particular project. In this paper we describe the first two layers, that is, the Feature Ontology, and the Customized Feature Ontology. In the following sub-sections we elaborate on these steps.

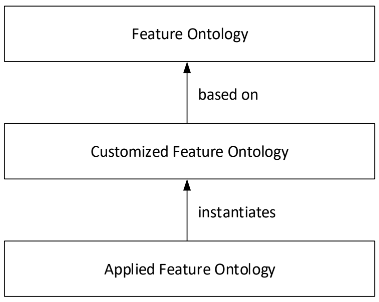


Fig. 8. The conceptual view for the adoption, customization and application of the selected feature ontology

A. ASELSAN SoS Metamodel

The ASELSAN Systems metamodel is shown in Fig. 9. Note that in this case a somehow different decomposition of the system elements is defined. This is based on the current practical view of the system decomposition. Hence the metamodel is also developed to reflect the practice.

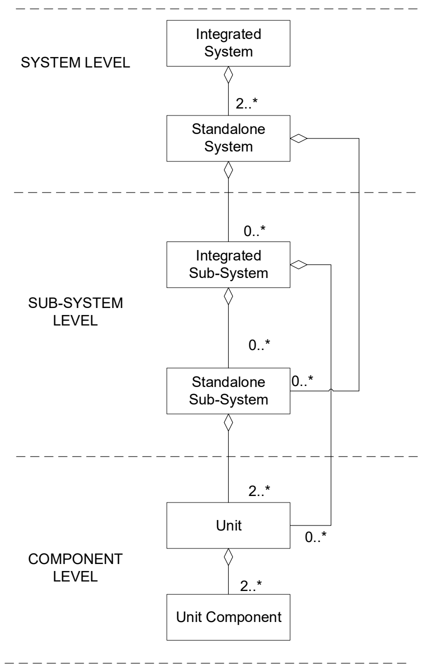


Fig. 9. Metamodel for ASELSAN SoS Structure

B. Customized Feature Ontology

Based on the metamodel for SoS, the ASELSAN Feature Metamodel is shown in Fig. 10.

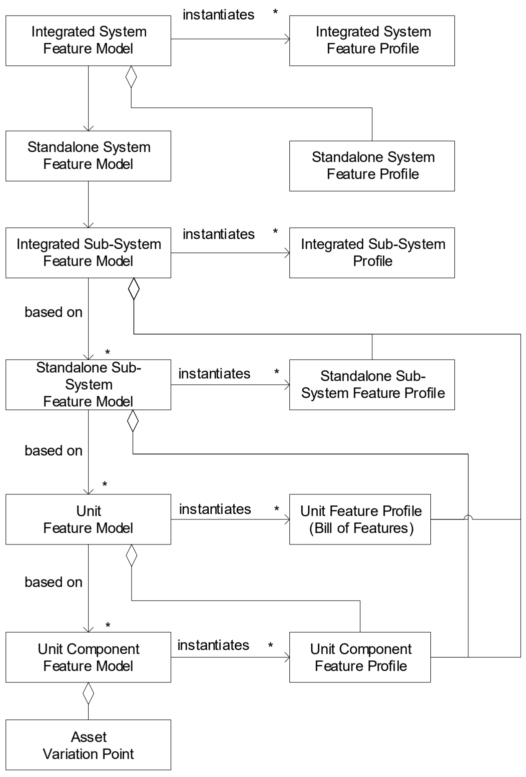

Integrated System Feature Model
instantiates *
Integrated System Feature Profile
Standalone System Feature Model
Standalone System Feature Profile
Integrated Sub-System Feature Model
instantiates *
Integrated Sub-System Profile
based on
*
Standalone Sub-System Feature Model
instantiates *
Standalone Sub-System Feature Profile
based on
*
Unit Feature Model
instantiates *
Unit Feature Profile (Bill of Features)
based on
*
Unit Component Feature Model
instantiates *
Unit Component Feature Profile
Asset Variation Point

Fig. 10. Metamodel for ASELSAN SoS-PLE Feature Modeling

C. Customizing Process for 2G-FPLE Process

In essence, the feature-driven PLE process can be applied in different ways. Fig. 11 shows the generic process flow for the 2G-FPLE. The arrows *Top-Down* and *Bottom-Up* shows the direction in which the process activities can proceed. A pure-top-down process will start with the first step that is *Develop Integrated System Feature Model* and continue until the last steps of Feature Modeling and System Architecture design processes. The bottom-up process will start with lower level units and define the variability and system modeling at increasing abstraction levels. For the ASELSAN Proline a top-down approach has been adopted which is shown in Fig. 12.

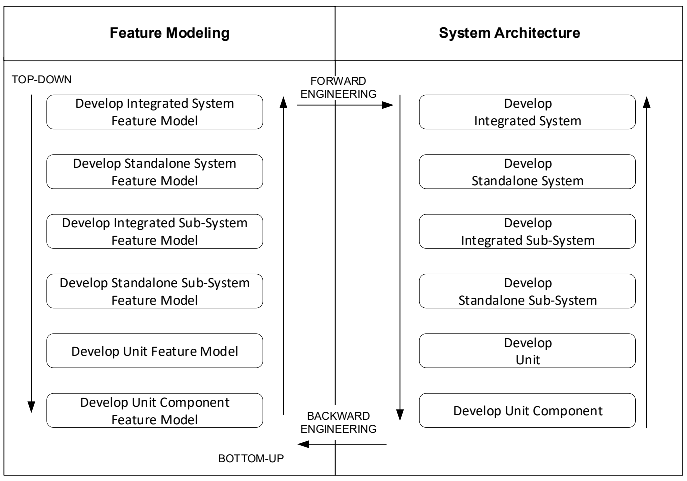

Feature Modeling
System Architecture
TOP-DOWN
FORWARD ENGINEERING
Develop Integrated System Feature Model
Develop Standalone System Feature Model
Develop Integrated Sub-System Feature Model
Develop Standalone Sub-System Feature Model
Develop Unit Feature Model
Develop Unit Component Feature Model
Develop Integrated System
Develop Standalone System
Develop Integrated Sub-System
Develop Standalone Sub-System
Develop Unit
Develop Unit Component
BACKWARD ENGINEERING
BOTTOM-UP

Fig. 11. Generic flow diagram for feature-driven PLE

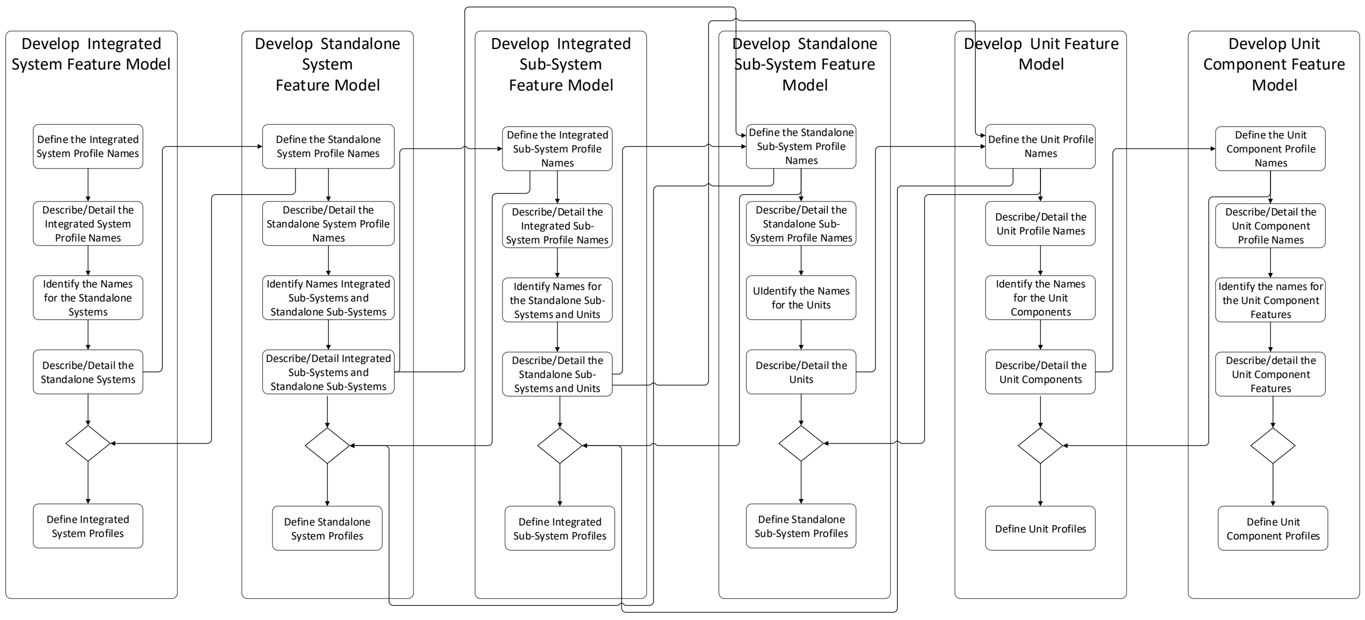

Develop Integrated System Feature Model
Define the Integrated System Profile Names
Describe/Detail the Integrated System Profile Names
Identify the Names for the Standalone Systems
Describe/Detail the Standalone Systems
Define Integrated System Profiles
Develop Standalone System Feature Model
Define the Standalone System Profile Names
Describe/Detail the Standalone System Profile Names
Identify Names Integrated Sub-Systems and Standalone Sub-Systems
Describe/Detail Integrated Sub-Systems and Standalone Sub-Systems
Define Standalone System Profiles
Develop Integrated Sub-System Feature Model
Define the Integrated Sub-System Profile Names
Describe/Detail the Integrated Sub-System Profile Names
Identify Names for the Standalone Sub-Systems and Units
Describe/Detail the Standalone Sub-Systems and Units
Define Integrated Sub-System Profiles
Develop Standalone Sub-System Feature Model
Define the Standalone Sub-System Profile Names
Describe/Detail the Standalone Sub-System Profile Names
UIdentify the Names for the Units
Describe/Detail the Units
Define Standalone Sub-System Profiles
Develop Unit Feature Model
Define the Unit Profile Names
Describe/Detail the Unit Profile Names
Identify the Names for the Unit Components
Describe/Detail the Unit Components
Define Unit Profiles
Develop Unit Component Feature Model
Define the Unit Component Profile Names
Describe/Detail the Unit Component Profile Names
Identify the names for the Unit Component Features
Describe/detail the Unit Component Features
Define Unit Component Profiles

Fig. 12. Adopted Top-Down Feature-Driven PLE Process

VI. Conclusion

In this paper we have reported on our experiences in customizing a feature ontology for a product line engineering process in a system-of-system context (SoS-PLE). For this we have analyzed the existing variability modeling approaches and selected the ontology as provided by the so-called Feature-Based Software and Systems Product Line Engineering ("Feature-Based PLE") which is the subject of an upcoming ISO standard that is in progress with involvement and support from INCOSE through its Product Line Engineering International Working Group. One of the key goals of the approach is to provide an enterprise ontology for features that is suitable for managing product line engineering in the largest and most complex product line organizations. This ontology appeared to be also useful for our own context, that is, a PLE process within an SoS context. Based on the inputs from the defined systems PLE and the feature-based PLE we have developed the customized PLE process. Hereby, we have developed the metamodel that represents the systems engineering concepts within ASELSAN and integrated this subsequently with the feature-based PLE approach. In contrast to the initial feature-based PLE approach we have developed a top-down process that starts with the configuration of the highest abstraction level which is incrementally refined to the lower more concrete levels until the eventual assets. The process has been developed to cope with the overall complexity and likewise realize the required scalability. In our future work we aim to validate the process for selected SoSs by applying the process in real projects.

References

[1] Aselsan website: http://www.aselsan.com.tr/default.asp?lang=en, accessed February 2011.

[2] G. Boeckle, P. Clements, J.D. McGregor, D. Muthig, & K. Schmid, K. Calculating ROI for Software Product Lines. IEEE Software 21, 3, pp. 23-31, June 2004.

[3] P.C. Clements, L. Northrop. Software Product Lines: Practices and Patterns. Boston, MA:Addison-Wesley, 2002.

[4] Guide to the Systems Engineering Body of Knowledge (SEBoK), October 2016

[5] M. Henshaw et al., "The Systems of Systems Engineering Strategic Research Agenda Systems of Systems Engineering," 8th Int. Conf. Syst. Syst. Eng. Maui, Hawaii, USA - June 2-6, 2013, no. 2, pp. 99–104, 2013.

[6] INCOSE Product Line Engineering International Working Group, http://www.incose.org/ChaptersGroups/WorkingGroups/analytic/product-lines, accessed October 2017.

[7] K. Kang, S. Cohen, J. Hess, W. Novak, S. Peterson, Feature Oriented Domain Analysis (FODA) Feasibility Study, Technical Report CMU/SEI-90-TR-021, 1990.

[8] C.W. Krueger, New Methods in Software Product Line Development. BigLever Software, Austin, TX; in Proc. of 10th Software Product Line Conference, 2006.

[9] C. Krueger, P. Clements. An Enterprise Feature Ontology for Feature-Based Product Line Engineering, 27th Annual INCOSE International Symposium (IS 2017). Adelaide, Australia, July 15-20, 2017.

[10] F. van der Linden, K. Schmid, E. Rommes. Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering, Springer, 2007.

[11] J.D. McGregor, S. Jarrad, L.M. Northrop, and K. Pohl. Initiating Software Product Lines, IEEE Software, vol. 19, no. 4, pp.24–27, 2002..

[12] K. Pohl, G. Böckle, F. van der Linden. Software Product Line Engineering – Foundations, Principles, and Techniques, Springer, 2005.

[13] K. Schmid, M. Verlage. *The Economic Impact of Product Line Adoption and Evolution*. IEEE Software, Vol. 19, No. 4, 50-57, 2002..

[14] M. Sinnema, S. Deelstra. Classifying Variability Modeling Techniques, Information and Software Technology, Volume 49 Issue 7, July, 2007.

[15] B. Tekinerdogan. Engineering Connected Intelligence: A Socio-Technical Perspective, technical report (inaugural lecture), Wageningen University, https://edepot.wur.nl/401115, 2017.