Context Modeling based on Feature Models Expressed as Views on Ontologies

Siniša Nešković*, Rade Matić**
* University of Belgrade/Faculty of Organizational Sciences, Belgrade, Serbia
** Belgrade Business School, Belgrade, Serbia
sinisa.neskovic@fon.bg.ac.rs, rade.matic@bbs.edu.rs

***Abstract*—This paper presents an approach for context modeling in complex self-adapted systems consisting of many independent context-aware applications. Contextual information used for adaptation of all system applications are described by an ontology treated as a global context model. A local context model tailored to the specific needs of a particular application is defined as a view over the global context in the form of a feature model. Feature models and their configurations derived from the global context state are then used by a specific dynamic software product line in order to adapt applications at the runtime. The main focus of the paper is on realization of mappings between global and local contexts. The paper describe an overall model architecture and provides corresponding metamodels as well as rules for mapping between feature models and ontologies.**

I. INTRODUCTION

Context-aware self-adapted systems (CASAS) are characterized by so called "smart applications" which react to users and their surroundings without user's explicit commands [1]. Such systems require an adaptation mechanism which timely adapts applications at the runtime according to changes in the context. The development of such mechanism is usually based on a single context model and rules that specify which configurations of the applications should run in every possible instance of the context [2, 3, and 4].

However, in case of very complex systems consisting of many context-aware applications, it is very difficult to use a single context due to its complexity. Such single global context must include all information needed by all applications, i.e. data about large number of different situations and different users with different interest and views. Thus, adaptation of a single working application must deal with the entire context including large amount of context data that are mostly irrelevant. On the other hand, a possible solution to this problem could be to use separate local contexts tailored for each particular application. However, such solution is also connected with many difficulties due to synchronization and potential inconsistencies among different local contexts possessing overlapping contextual information.

This paper presents an approach to the problem of self-adaptation in such complex CASAS which is based on usage of both global and local contexts. Global context is treated as an ontology describing contextual information required by all applications, whereas local contexts are derived as views over the global context tailored to the needs of particular applications. Views are defined through mappings (correspondences) between modeling elements of global and local contexts.

Additionally, local context models in our approach are expressed in the form of feature models (FM) [5], which are commonly used in software product line engineering (SPLE) to enable generation of application variants customized to specific needs of users. In our approach, derived feature models are used to instantiate variants of context-aware applications corresponding to a specific context state. Thereby, our approach relies on so called dynamic software product lines (DSPL) [6] as the main adaptation mechanism in CASAS.

The main focus of this paper is on describing the realization of mappings between global and local contexts. The rest of the paper is structured as follows. The next section gives an overview of work related to our research. In Section III our approach is described by an overall model architecture, brief description of the adaptation process and detailed descriptions of models used to realize global and local contexts. An example to illustrate our approach is also given. Final Section IV ends the paper with conclusions.

II. RELATED WORK

A lot of research has been recently dedicated to context modeling and development of context-aware systems. Several techniques are proposed for the representation of context [2, 7, and 8].

Ontologies are the most expressive and most used technique for modeling contextual information [9, 10, 11, and 12]. The application of ontologies for context modeling provides a unique way to specify key concepts as well as a number of additional concepts and instances, and thus allows reuse and sharing of information or knowledge about the context in distributed systems. However, limitations of this technique are various [7, 13, 14, and 15]. Most of suggested ontologies do not provide a clear description of contextual information. Different ontologies have been proposed to model domain specific context information or generic models reusable in many domains. All of them have certain drawbacks in generality and/or dynamicity [4]. General problem in all ontologies is that suggested model fails in using and presenting generic context ontology because they may contain useless ontologies in specific applications, which limits their usage and extensibility. For example, location ontology is worthless in a context-aware system operating in local area.

Approaches defined in [4, 16, 17, and 18] use feature modeling as a technique for context modeling and development of context-aware application in order to

improve reusability and new configurability. Feature models offer different degree of formalism and expressiveness. Looking at the practical applicability and usability, which are not discussed here in detail, it can be stated that the more extensions for feature models are used the more practical and usable for context modeling it is. A limitation of feature models is that concepts or relations are defined in an unsuitable way. Feature models do not have a clear view of the relation between contexts and features. It is thus difficult to determine if the features in a feature model are arranged and structured consistently with domain knowledge and whether they are accurately expressed, organized and represented.

Hybrid models represent a combination of two or more different modeling techniques for different usage, either general or domain-specific. In order to gain more beneficial, flexible and general applications, many researchers try to integrate or expand various context modeling techniques [19 and 20].

Surveys of self-adaptation software are presented in [21 and 22], but they do not analyze SPL as software adaptation approaches. Other approaches that apply SPLE paradigm to develop adaptive systems use a feature model for variability [23, 24, 25, 26, 27, and 28]. UbiFEX was presented by Fernandes et al. [16]. Their aim is to provide a modeling notation that extends feature models with context feature models. In [16, 18] context-aware adaptation is discussed. Desmet et al. [29], propose Context-Oriented Domain Analysis (CODA) while Hartmann et al. [18] present high-level context information using context variability models (CVM). Using reconfiguration patterns which are based on UML collaboration and state diagrams, Gomaa et al. [30] propose a solution for dynamic reconfiguration of SPL. The paper presented by Ref. [4] is very interesting and similar to our research. Despite the similarities, our paper differs in the main idea of mapping ontology with FM and making views on ontology. Regarding adaptation mechanisms several research projects, such as MADAM [31], MUSIC [32], DiVA [33] and Trinidad et al. [26] address component-based architecture to provide development of self-adaptive systems. To control adaptation, MADAM use architectural models and SPL techniques, at runtime. MUSIC continued on the works of the MADAM project developing methodology and tools for adaptation of mobile application. Compared to DiVA, the main difference is in implementations for each component type of the architecture. Ref [26] assumes that feature represent component enabling DSPL to dynamically include or exclude its components at runtime.

III. OUR APPROACH

This section describes our approach to context modeling. We first give an overall model architecture, which identifies all models (including their metamodels) and their relationships required in our approach. We also briefly describe the adaptation process and how ontology models can be mapped to feature models.

A. Model architecture

The overall model architecture is shown as an UML package diagram in Fig. 1.

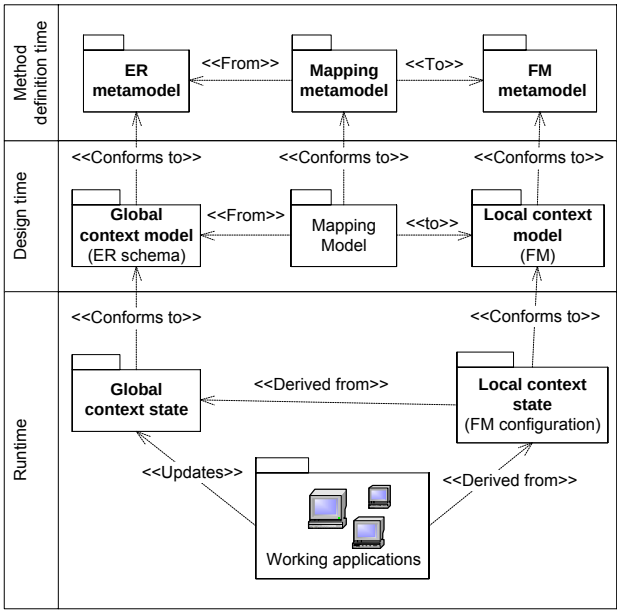

Figure 1. Model architecture

UML packages in the diagram represent models, whereas various relationships among models are represented by stereotyped dependency associations between corresponding packages. The diagram also classifies models in three different categories according to the time when they are created (represented as swim lanes in the diagram):

- *Runtime* category encompasses models which are created during runtime of CASAS components and user applications.
- *Design time* category encompasses models which are created during the design of CASAS components and user applications.
- *Method definition time* category encompasses models which are defined by our approach, i.e. metamodels which are introduced in the next subsection of the paper.

In our approach the Entity-Relationship (ER) data model is used as an ontology definition language [34, 35, 36, and 37]. Hence, a global context model is defined using an ER schema, which must conform to ER metamodel. On the other hand, feature models are used to represent local context models which must conforms to the FM metamodel.

Feature models are defined as views on ontologies, namely, as projections of the ontologies from different viewpoints [38]. The views definition is defined by a mapping model which maps concept of an ER schema to concepts of a FM. The defined mappings have to follow rules and constraints, which are defined by the Mapping metamodel.

At the runtime level, a CASAS maintains a global context state, which keeps contextual information at the particular moment. The global context state is usually realized as some form of a database structured according to its ER schema defined in the design time. The database is updated, i.e. the global context state is maintained, by context-aware applications and other CASAS's run time components.

When significant changes of a context are detected, CASAS run time components will trigger an adaptation process which will instantiate (generate) affected running applications. The adaptation process consists of two main steps:

- ***Derivation of local contexts***. Local context states for affected running applications are derived from the current global context state using the corresponding mapping models defined at the design time. According to SPL engineering principles, the local context state is represented as a FM configuration.
- ***Instantiation of running applications.*** Using a DSPL, affected working applications are instantiated based on corresponding FM configurations. Thus, a new instance (version) of the application is adapted to the current local context state.

The adaptation process is preformed by a part of CASAS called Adaptation Manager. Due to space limitations, detailed description of the adaptation process and Adaptation Manager implementation is not included here.

B. Metamodels and mapping rules

ER and FM metamodels as well as the Mapping model are shown in Fig. 2. ER metamodel is based on ER model defined in [39]. ERConcept represent the most abstract concept in the ER data model. It is specialized into more concrete ER concepts:

- *Entity* represents types of objects in a system. It is further specialized into Kernel, Subtype, Aggregation, and Week entity types.
- *Relationship* between two entity types.
- *Mapping* which represents relationship roles as well as special relationships between specific entity types. *Min* and *Max* attributes specify lower and upper bound of its cardinality. *Mapping* is further specialized into more concrete subtypes: *OrdinaryMapping* (i.e. relationship role), *WeakMapping*, *AggregationMapping*, and *SpecializationMapping*.
- *Attribute* describes an entity type and *Domain* specifies the type value for an attribute.

The FM metamodel shown in Fig. 2 is an original version developed by authors independently of other FM metamodels available in the literature. Its most abstract concept is FMConcept which is further specialized into more concrete concepts:

- *Feature* can be either solitary feature, grouped feature or feature group. *Feature attribute* is also defined as a specialized type of feature.
- *Relationship* which represents an association between two features. It can be usual hierarchical feature/subfeature association, but also an association between feature group and grouped features. *FDreference* represent a reference to another FM enabling a division of large feature models into smaller ones.

The Mapping metamodel defines allowed correspondences between ER concepts and FM concepts. Allowed correspondences are determined by the following rules:

- An ER schema maps to a Feature model.
- Each *Entity* type (*Kernel*, *Subtype*, *Aggregation*, and *Week*) maps to a Feature.
- Each *Attribute* and *EnumerationLiteral* maps to a Feature.
- Each *SpecializationMapping* maps to a *GroupedRelationship* where Supertype in this mappings maps into a Feature whose attribute "IsFeatureGroup" is set to true. Their subtypes become features whose attribute *IsGroupedFeature* is set to true.
- Each *WeekMapping*, *AggregationMapping* and *OrdinaryMapping* maps to a Single Relationship.
- Each *Domen* maps to a TypeAtribute.
- Cardinality of any mapping becomes cardinality of relationships of their corresponding features.
- Mandatory of a feature depends from the cardinality of mapping.
- When two entities have two or more relationship or if tree or more entities form a circle than *FMReference* is created.

C. An example

In this section we illustrate our approach by an example of a CASAS aimed to support a consortium of flower stores in a big city. The consortium has made an agreement with local taxi drivers to deliver flowers from the stores to their customers. When a store gets a flower delivery order from a customer, it creates a request which is sent to drivers from the store in order to select a driver assigned for the actual delivery. Drivers compete for the delivery by sending their current location. Depending of preferences of stores (e.g. automatic or manual delivery assignments to drivers, delivery confirmation required or not, etc.) and equipment options for drivers (e.g. whether driver is equipped with a GPS device), there can be many different variants of applications supporting stores and drivers. Here we give three use cases (UC) with contextual variants:

1. **Use case:** *Select driver for delivery*
   Actor: *Florist*
   - The system sends offer for delivery to all drivers
   - The system registers the positive responses and the location of the driver
   - The system ranks all driver responses based on current driver distances from the store
   - Variant 1: Automatic Assignment
     - The system selects driver with the best rank.
   - Variant 2: Manual assignment
     - Florist manually selects the driver from the driver ranking list.
2. **Use case:** *Send bid for delivery*
   Actor: *Driver*
   - The driver receives a request for a delivery
   - The driver accepts the offer
   - Variant 1: driver with GPS device
     - The system gets the current location from the driver's GPS device
   - Variant 2: driver without GPS device
     - Driver enters current location
   - Send confirmation response and current location

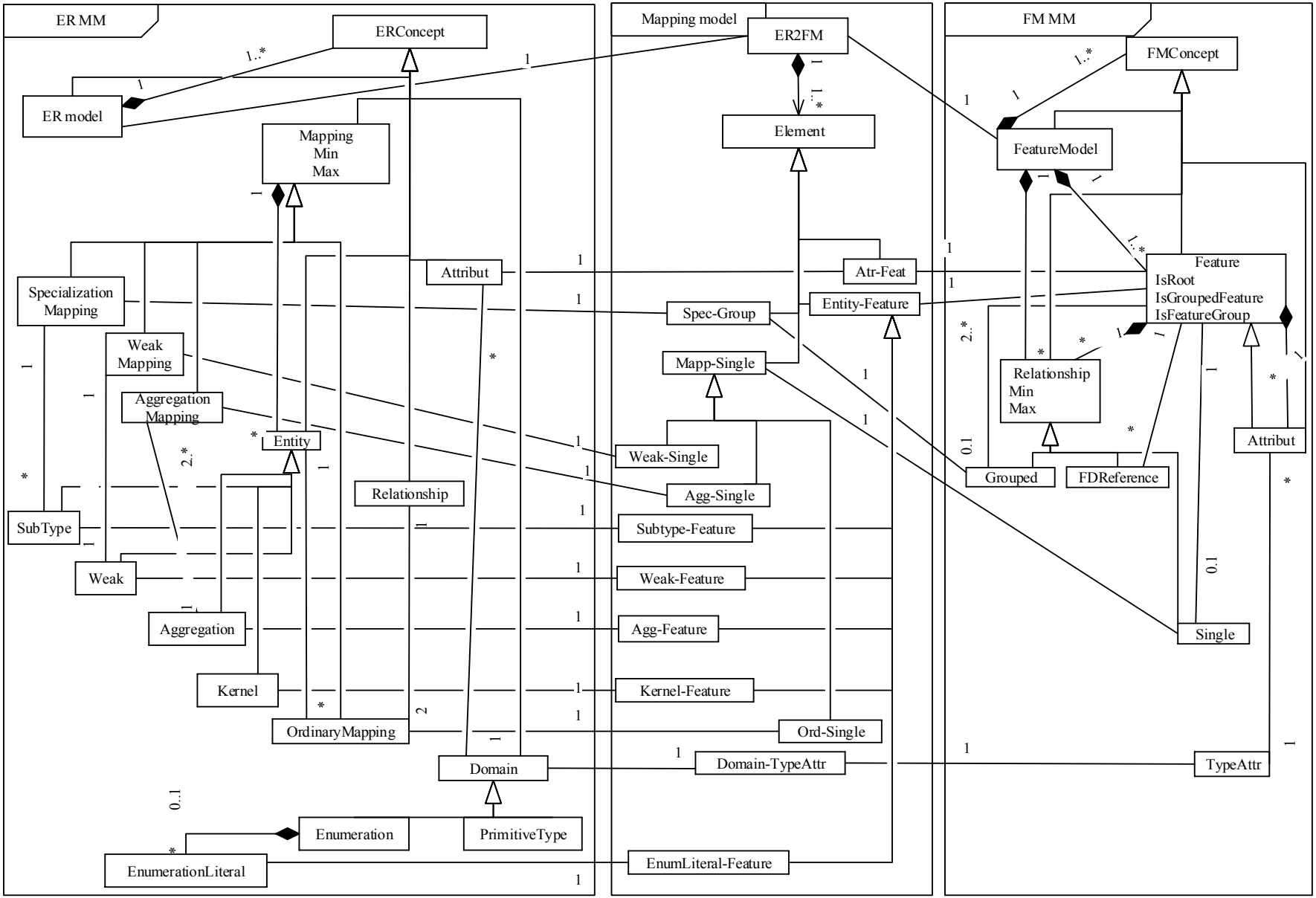

Figure 2. Mapping metamodel

3. Use case: *Confirming delivery*

Actor: *Driver and Customer*

- Variant 1: Store requires delivery confirmation
  - The driver asks the client to enter confirmation code given to him buy the store
  - Confirmation code is sent to the store
- Variant 2: Store doesn't require confirmation
  - The driver has no this use case

In order to do required adaptations, our CASAS system must keep the contextual information about stores and drivers. Based on the given use case variants, the (simplified) global context model is given in Fig. 3.

Since stores and drivers has two independent applications for realizing appropriate use cases, each application must have its own local context model. The two appropriate local context models expressed as feature models are given in Fig. 4.

On the left side of Fig 4. is a FM for the application supporting store (UC 1) is shown. On the right is a FM supporting taxi driver use cases (UC 2 i UC 3). Both models include only relevant contextual informatuon projected from the global context. For example, FM on the left doesn't have information about drivers, because the adaptation of store application does not depend on such contextual information.

Due to space limitations, the corresponding mapping models between the global context model and the two local context models are not given here. For the same reason the corresponding FM configurations are also not given here.

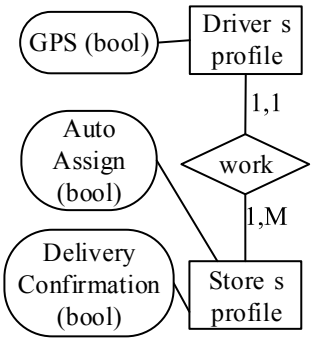

Figure 3. An example of global context model

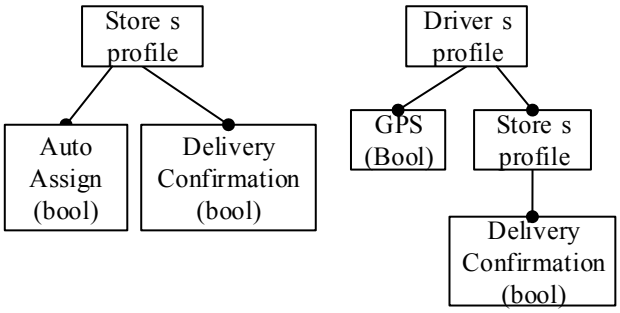

Figure 4. An example of two local context models

IV. CONCLUSIONS

The main advantage of our approach stems from the utilization of both global and local contexts modeled by two different modeling techniques. Ontologies are superior for context modeling and realization of global context state, but not so suitable for the adaptation purposes in DSPL. On the other hand, feature models are suitable for the adaptation purposes, but inadequate for global context modeling. Thus, our approach takes the best of both ontologies and feature models by using synergy effects.

Comparing to other existing approaches, the key benefit of our approach is in the adaptation process. It can be much more efficient due to smaller, less complex and better tailored local context models. This efficiency is achieved without sacrificing the advantages of ontologies as a superior knowledge representation technique for context modeling.

REFERENCES

[1] A.Schmidt, "Implicit human-computer interaction through context," 2nd Workshop on Human Computer. Interaction with Mobile Devices, 1999.

[2] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems," Int. Journal of Ad Hoc and Ubiquitous Computing, 2(4):263–277, June 2007.

[3] C. Bolchini, C. A. Curino, E. Quintarelli, F. A. Schreiber , L. Tanca, "A data-oriented survey of context models". ACM SIGMOD Record, vol.36 no.4, December 2007.

[4] Z. Jaroucheh, X. Liu, and S. Smith, "CANDEL: Product Line Based Dynamic Context Management for Pervasive Applications," International Conference on Complex, Intelligent and Software Intensive Systems (ARES/CISIS 2010), IEEE CS, 2010, pp. 209-216.

[5] K. Kang, S. Cohen, J. Hess, W. Novak, and S. Peterson, "Feature-Oriented Domain Analysis (FODA) Feasibility Study," Software Engineering Institute, Carnegie Mellon University,Tech. Rep. CMU/SEI-90-TR-21, Nov. 1990.

[6] S. Hallsteinsen, M. Hinchey, S. Park, and K. Schmid, "Dynamic Software Product Lines, " Computer, 41(4):93–95, 2008.

[7] C. Bettini , O. Brdiczka , K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, D. Riboni,"A survey of context modeling and reasoning techniques, Pervasive and Mobile Computing," vol.6 no.2, p.161-180, April, 2010.

[8] T. Strang and C. Linnhoff-Popien,"A context modeling survey," In 1st Int. Workshop on Advanced Context Modelling, Reasoning and Management, 2004.

[9] H. Chen, T. Finin, and A. Joshi, "An Ontology for Context-Aware Pervasive Computing Environments," The Knowledge Engineering Review, vol. 18, pp. 197-207, 2004.

[10] X. H. Wang, T. Gu, D. Q. Zhang, and H. K. Pung, "Ontology Based Context Modeling and Reasoning using OWL," presented at Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004.

[11] T. Strang, C. Linnhoff-Popien, and K. Frank, "CoOL: Context Ontology Language to enable Contextual Interoperability," Lecture Notes in Computer Science, vol. 2893, pp. 236-247, 2003.

[12] [23X] F. Fuchs, I. Hochstatter, M. Krause, and M. Berger, "A Metamodel Approach to Context Information" presented at Third IEEE International Conference on Pervasive Computing and Communications Workshops, 2005.

[13] J. Zakwan, L. Xiaodong and S. Sally, "Mapping features to context Information: supporting context variability for context-aware pervasive applications". International Joint Conference on Web Intelligence and Intelligent Agent Technologies (WI-IAT 2010). IEEE Computer Society, Toronto, Canada, pp. 611-614, August 2010.

[14] Vanathi, B. and V.R. Uthariaraj, "Hybrid hierarchical context representation in a context aware system," in Proc. of the 2nd

International Conference on IT and Business Intelligence (ITBI'10), IEEE and IEEE Computational Intelligence Society, Nagpur, 2010.

[15] Vanathi, B. and V.R. Uthariaraj, "Collaborative Context Management and Selection in Context Aware Computing," in Communications in Computer and Information Science, 1, Vol. 133, Advanced Computing, Part 4, Springer-Verlag Berlin Heidelberg, pp: 348-357.

[16] P. Fernandes, C. Werner, E. Teixeira, "An Approach for Feature Modeling of Context-Aware Software Product Line," Journal of Universal Computer Science, Special Issue on Software Components, Architectures and Reuse, vol. 17, no. 5, pp.807-829, 2010.

[17] M. Acher, P. Collet, F. Fleurey, P. Lahire, S. Moisan, J.P. Rigault, "Modeling Context and Dynamic Adaptations with Feature Models," in Int'l Workshop Models@run.time at Models 2009 (MRT'09). October, 2009.

[18] Hartmann, H., Trew, T., "Using Feature Diagrams with Context Variability to Model Multiple Product Lines for Software Supply Chains," in 12th International Software Product Line Conference, IEEE (2008), pp. 12-21, Ireland, September 2008.

[19] K. Henricksen, S. Livingstone, and J. Indulska," Towards a hybrid approach to context modelling, reasoning and interoperation," Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management, 2004.

[20] I. Roussaki, M. Strimpakou, N. Kalatzis, M. Anagnostou, and C. Pils, "Hybrid context modeling: A location-based scheme using ontologies," PerCom Workshops, IEEE Computer Society, 2006.

[21] M. Salehie, L. Tahvildari, "Self-adaptive software: landscape and research challenges," ACM Transactions on Autonomous and Adaptive Systems 4, 2009.

[22] K. Kakousis, N. Paspallis, G.A. Papadopoulos, "A survey of software adaptation in mobile and ubiquitous computing," Enterprise Information Systems (UK) 4, 355–389, 2010.

[23] C. Cetina, P. Giner, J. Fons, and V. Pelechano, "Using Feature Models for Developing Self-Configuring Smart Homes," in Proc. of Int'l. Conf. Autonomic and Autonomous Systems (ICAS), pages 179–188. IEEE CS, 2009.

[24] S. Hallsteinsen, E. Stav, A. Solberg, and J. Floch, "Using Product Line Techniques to Build Adaptive Systems, " in Proc. Int'l. Software Product Line Conf. (SPLC), pages 141–150. IEEE CS, 2006.

[25] J. Lee and K. C. Kang, "A Feature-Oriented Approach to Developing Dynamically Reconfigurable Products in Product Line Engineering," in Proc. Int'l. Software Product Line Conf. (SPLC), pages 131–140. IEEE CS, 2006.

[26] P. Trinidad, A. Ruiz-Cort´es, and J. Pe˜na, "Mapping Feature Models onto Component Models to Build Dynamic Software Product Lines, " in Int'l. Workshop on Dynamic Software Product Lines (DSPL), pages 51–56. Kindai Kagaku Sha Co. Ltd., 2007.

[27] [M2011Y] M. Rosenmüller , N. Siegmund , M. Pukall, S. Apel, "Dynamic software reconfiguration in software product families, " in Software Product-Family Engineering, " Lecture Notes in Computer Science, 2004.

[28] B. Cheng, R. de Lemos, H. Giese, P. Inverardi, and J. Magee, editors, " Software Engineering for Self-Adaptive Systems, " vol. 08031 of Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrumfuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2008.

[29] B. Desmet, J. Vallejos, P. Costanza, W. De Meuter, and T. D'Hondt, "Context-Oriented Domain Analysis, " in 6th International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT 2007), Lecture Notes in Artificial Intelligence. Springer-Verlag, August 2007.

[30] H. Gomaa and M. Hussein. Dynamic software reconfiguration in software product families. In Software Product-Family Engineering, Lecture Notes in Computer Science, 2004.

[31] K. Geihs et all, "Software engineering for self-adaptive systems," Springer-Verlag, Berlin, Heidelberg, Chapter Modeling of Context-Aware Self-Adaptive Applications in Ubiquitous and Service-Oriented Environments, 2009.

[32] S. Hallsteinsen , K. Geihs , N. Paspallis , F. Eliassen , G. Horn, J. Lorenzo , A. Mamelli , G. A. Papadopoulos, "A development framework and methodology for self-adapting applications in ubiquitous computing environments, " Journal of Systems and Software, v.85 n.12, p.2840-2859, December, 2012.

[33] B. Morin, O. Barais, J-M. Jézéquel, F. Fleurey, and A. Solberg, "Models@ run. time to support dynamic adaptation," Computer 42, no. 10: 44-51, 2009.

[34] V. Devedžić, "Understanding ontological engineering," Communications of the ACM, vol.45 no.4, April 2002.

[35] D.M. Sanchez, J.M. Cavero, and E.m. Martinez, "The road toward ontologies," in ONTOLOGIES : A handbook of priciples, concepts and applications in information systems, R. Sharman, R. Kishore, and R. Ramesh, Eds. London: Springer, 2006, pp. 3-20.

[36] M. Jarrar, J. Demey, R. Meersman, "On Using Conceptual Data Modeling for Ontology Engineering," in Spaccapietra, S., March, S., Aberer, K., (Eds.): Journal on Data Semantics (Special issue on Best papers from the ER, ODBASE, and COOPIS 2002 Conferences). LNCS, Vol. 2800, Springer, pp.:185-207. October 2003.

[37] K. Rajiv Kishore, Z. Hong , R. Ramesh, "A Helix-Spindle model for ontological engineering," Communications of the ACM, vol.47 no.2, p.69-75, February 2004.

[38] K. Czarnecki, P.K. Chang Hwan, K.T. Kalleberg, "Feature Models are Views on Ontologies," proceedings of the 10th International on Software Product Line Conference, p.41-51, August 21-24, 2006.

[39] B. Lazarević, Z. Marjanović., N. Aničić, S. Babarogić, „Baze podataka", Beograd, 2006.