Semana de Engenharia 2010
Guimarães, 11 a 15 de Outubro

An Ontology-based Approach to Model-Driven Software Product Lines

Nuno Ferreira
Dep. Sistemas de Informação
Universidade do Minho
nuno.ferreira@dsi.uminho.pt

KEYWORDS
Software Engineering Process; Software Design.

ABSTRACT

Software development in highly variable domains constrained by tight regulations and with many business concepts involved results in hard to deliver and maintain applications, due to the complexity of dealing with the large number of concepts provided by the different parties and system involved in the process. One way to tackle these problems is thru combining software product lines and model-driven software development supported by ontologies. Software product lines and model-driven approaches would promote reuse on the software artifacts and, if supported by an ontological layer, those artifacts would be domain-validated. We intend to create a new conceptual framework for software development with domain validated models in highly variable domains. To define such a framework we will propose a model that relates several dimensions and areas of software development thru time and abstraction levels. This model would guarantee to the software house traceability of components, domain validated artifacts, easy to maintain and reusable components, due to the relations and mappings we propose to establish in the conceptual framework, between the software artifacts and the ontology.

INTRODUCTION

Highly variable business domains (like insurance) faces many and specific problems, that derives from the weight of legacy technologies in the information system, thru the way the contact with the outside world is made, passing by domain-specific variability. Applications are traditionally built in a straightforward way, many times having no relation whatsoever between them. When dealing with legacy systems that problem emerges even more, due to the time elapsed between the requirements specification for each developed module and the present. Usually, each application has its own configuration, store its own data and is guided by its own business rules (Simonov, Soroldoni et al. 2004).
One of the major costs in software is its maintenance, as it becomes more complex and expensive thru time. In highly dynamic domains, where a great maintenance effort is needed to keep the software up-to-date, the overall cost rises and the time-to-market is critical, there is a need to improve the development process and manage the variability and reusability. One of the key challenges is to understand and identify the relations and representations of software artifacts and resources involved in the maintenance
To address the previously exposed problem, the current proposal is to deliver a conceptual framework to the software development based on models in a product line scenario supported by ontologies. Combining software product lines with a model-driven approach, a highly-customizable, reusable and generative software development environment can be achieved. The desired solution is the integration of ontologies for the purpose of more effective model-driven development in insurance product lines.

RESEARCH OBJECTIVES AND APPROACH

As a main research objective it has been defined the formalization of an approach that combines multi-stage (time-variant stages), with ontological support and multi-level primitives (abstraction levels) for the insurance domain software process development. Multi-stage allows the mapping of multi-level primitives into the stages. The insurance ontological reference is able to unify that mapping. Stages are also useful to delimiter borders and make artifacts as deliverables between stages (Bragança and Machado 2008). Each stage can be viewed as a separate company, delivering products to the next stage and receiving what they need from the previous ones. Each one has its one reality, characteristics and support, all disconnected from the others. This way we can only pass artifacts and documentation between stages, allowing, for instance, the software house to actively communicate with the insurance company and vice-versa.
Ontologies will act as a guideline containing the core business and development concepts required by the mode-driven tools to generate specialized and business validated software artifacts. Those artifacts will be traceable along the development lifecycle, from the core assets thru the final product installation and configuration in the customer. Taking into account that the software development process relies on a product line, assets reusability and categorization would be guaranteed by the links established between the artifacts and the ontology.
It will be proposed a process that aims mapping the inherent temporal dimension in product lines with the artifacts' abstraction levels. That mapping can rely on insurance ontologies to guarantee its coherence and act as a guideline.

CURRENT WORK

In the on-going work, domain of insurance is being investigated. This domain is used as an application baseline, which will first help to extract specific needs coming from the real world, and finally, it will help to evaluate the achieved results. The insurance domain deals with business and technological variability in various aspects. Its business models depend upon product line diversity, commercialization channels, or governmental or institutional laws. There are many different product lines in this specific domain, namely Life, Health, Group, Intermediary Property and Casualty products. It must be accounted that many back office management activities are executed by stable and large legacy systems. Customers interact with those systems through internet and employees through the intranet of the company (Simonov, Soroldoni et al. 2004). As the information demand increases, also increases time taken to provide solutions (Chris van 2005).

Our current approach aims at mapping ontologies to the software product line development process in a model-driven approach. Regarding Figure 1, the multi-stage ontological support (horizontally defined) allows the mapping of muli-level primitives (vertically placed) into the stages. The ontological referential is able to unify that mapping. Stages are also useful to delimiter borders and make artifacts as deliverables between stages (Bragança and Machado 2008). Each stage can be viewed as a separate company, delivering products to the next stage and receiving what they need from the previous ones. Each one has its one reality, characteristics and support, all disconnected from the others. A stage is directly related to time, being an object that has its own lifecycle. Stages may begin with the initial phase, where there is no core assets defined in the domain level. If there are core assets defined, stages automatically begin in a posterior stage. A level refers to abstraction, being the first level the most concrete, where all artifacts are implemented and executed. The last level is the most abstract, usually the building plan that guides the core assets. Some of the needed transformations will be later formalized using QVT (OMG 2009).

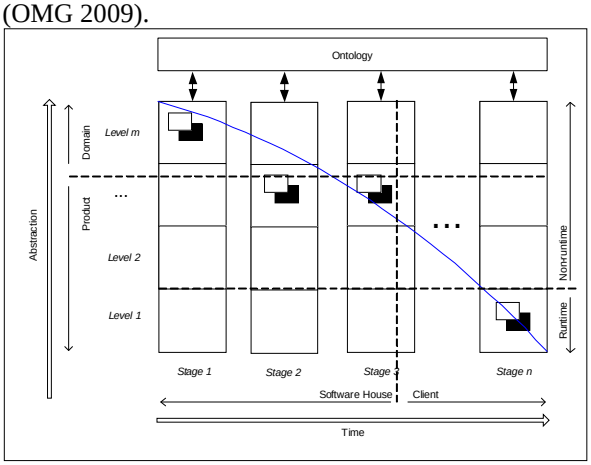

Figure 1: Multi-stage ontologically-sustained approach to SPL multi-level architectures

We have represented the “domain” section, which refers to the domain engineering level or core assets development, where the basis for the SPL are developed, in an organized fashion. Also represented is the “product” section, referring the application engineering level, or product development, where the assets previously defined are put together to work. Two other dimensions are the “client” and the “software house”. Regarding the software house, it represents the place where the artifacts are being developed and by client we refer to the customer where the product will be running. Also related to the previous dimensions we have run-time and non run-time. By non-runtime we mean the levels where the artifact isn’t executable. By run-time, it is meant the levels where the artifact is executing.

CONCLUSION

In this paper we related our proposal consisting on adding two main dimensions to the product line approach, time and abstraction. If so, the degree of conceptualization rises. A multi-stage process approach is required to deal with the time elapsed in the different product development phase, from core asset thru final product, executed in the customer’s environment. The other dimension, the degree of abstraction referred as multi-stage, implies different conceptualizations of the system.

An ontological approach is necessary to unify and guide the process which coordinates all the abstraction and time of the development lifecycle in a product line. We are trying to create a process that manages the evolution of software product line artifacts over time and ensures the consistent integration of the changes in all affected product line applications.

REFERENCES

Bragança, A. and R. J. Machado (2008). Transformation Patterns for Multi-staged Model Driven Software Development. 12th International Software Product Line Conference - SPLC 2008. Limerick, Ireland, IEEE Computer Society Press, Los Alamitos, California, U.S.A.**:** 329-338.

Chris van, A. (2005). Organizational Principles for Multi-Agent Architectures (Whitestein Series in Software Agent Technologies), Birkhauser.

OMG. (2009). "QVT 1.0." from http://www.omg.org/spec/QVT/1.0/.

Simonov, M., M. Soroldoni, et al. (2004). Ontology-driven Natural Language access to Legacy and Web services. BIS2004.