Towards the Generation of Interoperability Connectors using Software Product Line Engineering

Boubou T Niang, Giacomo Kahn, Nawel Amokrane, Yacine Ouzrout, Mustapha Derras, Jannik Laval

Towards the Generation of Interoperability Connectors using Software Product Line Engineering

Boubou T. Niang[1,2], Giacomo Kahn[1], Nawel Amokrane[2], Yacine Ouzrout[1], Mustapha Derras[2], and Jannik Laval[1]

[1] Univ Lyon, Univ Lumière Lyon 2, INSA Lyon, Université Claude Bernard Lyon 1, DISP, EA4570, 69676 Bron, France prenom.nom@univ-lyon2.fr
[2] Berger-Levrault prenom.nom@berger-levrault.com

Abstract

Information Systems (IS) of modern companies must be reactive and capable to communicate with third-party IS and within their subsystems. However, systems and applications are independently designed and conform to different technical and domain standards that are continuously evolving. In this heterogeneous context, the communication between systems is ensured by interoperability connectors whose properties vary according to the communication needs. Thus, development and maintenance of the connectors can therefore be tedious. This paper presents an ongoing research project to automatically generate all or a part of interoperability connectors. To do this, we combine a software product line approach and model-driven engineering techniques.

Abstract

Les systèmes d'information (SI) des entreprises modernes doivent être réactifs et capables de communiquer avec des SI tiers tout en guaratissant la communication entre les applications internes. Cependant, les systèmes et les applications sont conçus de façon independante et sont conformes à différentes normes en constante évolution. Dans ce contexte d'hétérogénéité, la communication entre les systèmes est assurée par des connecteurs d'interoperabilité dont les propriétés varient en fonction des besoins de communication. Ainsi, le développement et la maintenance des connecteurs peuvent être fastidieux. Cet article présente une recherche en cours pour la génération automatique de tout ou d'une partie des connecteurs d'interoperabilité. Pour ce faire, nous combinons l'approche des lignes de produits logiciels et des méthodes d'ingénierie dirigée par les modèles.

1 Introduction

The Information System (IS) occupies an important place in the proper functioning of modern companies. Indeed, companies need to collaborate with many partners, which are structurally diversified and operate in various business sectors. Despite this diversity, the IS of the various stakeholders and their internal software must be able to communicate and work together in a natural way as if they were one system of Information Systems (SoIS) [13] to enable a successful collaboration. For this, it is necessary to enable interoperability between subsystems, at least for data exchanges. These exchanges are ensured by architectural elements that manage constituent interactions, called connectors. Connectors are first-class entities of the architecture that describe communication semantics between heterogeneous constituents. It can also activate the behavior of the provided interfaces at runtime [10]. The development and updating of connectors can be tedious and very time consuming as they have characteristics depending on the need for interoperability, e.g. communication protocol, data format, data type. To facilitate the evolution of interoperable systems, several studies have been carried out for the

rapid implementation of interoperability connectors [12] [16] [4]. These works on connectors consider them as independent constituents of business applications. This exogenous view of the connector favors the decoupling of the information system components. The decoupling of business applications of the SoIS enable to delegate the choice of requested external service to the connector and to make it configurable. Another interest of decoupling is that the evolution of exchange formats is supported by the connectors not by the communicating applications, avoiding this way to impact the latter. Moreover, decoupling has the advantage of avoiding the direct connection and direct call between business applications, as these types of communication are sometimes unsecured or impossible. We therefore consider connectors as separate yet integral constituents of the IS. Their characteristics and function as the carriers of data exchanges can be generalised to a certain extent allowing as to examine the possibility to automatically generate all or a part of their code.

This paper presents an ongoing research project that aims at facilitating the scalability of the ISs by acting on the exchange system to enable configuration and automatic generation of all or a part of the interoperability connectors. For this purpose, the set of connectors is considered as a software product line (SPL) [5] to easily manage variability. The approach will also exploit the model-driven engineering (MDE) [15] approach to better manage complexity of the connectors' generation system.

2 The SPL and MDE combination approach

A Software Product Line (SPL) is defined as "a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segmentor mission and that are developed from a common set of core assets in a prescribed way" [5]. The SPL engineering process consist in two phases: Domain Engineering (DE) and Application Engineering (AE). The DE sub-process consists in specifying artefacts for reuse and the AE sub-process consists in deriving one product by reuse [15]. The proposed approach is based on this same mechanism.

In the DE sub-process, the specifications and source codes of the existing connectors are analyzed. Through this analysis, we extract the connectors' characteristics and represent their commonalities and variability in a feature model (FM) that represents the configurability aspect of reusable software at an abstract level [8]. Then, the FM is used to build artefacts for reuse : APIs related artifacts such us provided resources or client calls, code support for exchanged data serialisation (model or domain classes), or connection configuration to a queuing system.

The AE sub-process consist first in configuring the FM according to the specification of a desired connector. Thus, the configured FM is transformed into a connector model using a model-to-model transformation techniques. To finish, the model-to-text transformation methods is applied to the resulting model to generate the source code of the specified connector. Figure 1 shows the process.

The proposal combines a SPL approach and model-driven engineering (MDE) methods. While SPL allows to manage variability, MDE helps to reduce the gap between problem and implementation domain. Thus, The transition from FM to source code goes through an intermediate model. This enable flexibility and makes the generation system platform independent. In the DE sub-process, various tasks are performed manually once, but may change occasionally if the requirements of the domain change. Except the specification step, each stages in AE phase are performed automatically to generate a desired connector.

To implement the process, we have to face some existing obstacles encountered in SPL engineering. Some approaches have tackled these different barriers [7] [1] [2] [6] [11]. In [7] the

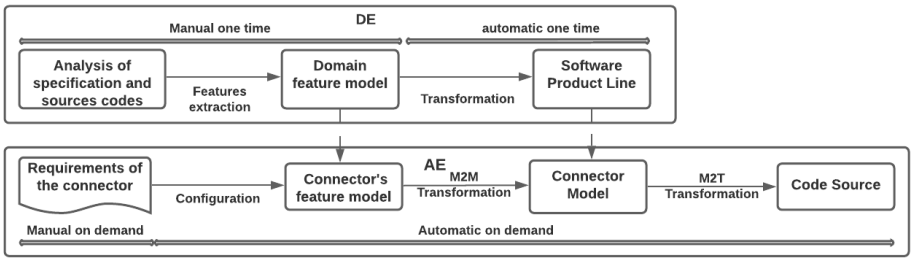

Figure 1: Overview of the proposed SPL approach consisting in two sub-processes DE and AE.

authors proposes a staged configuration where each step is tailored for a specific stakeholder. Reference [2] develop an automated approach for selecting a set of features that would satisfy both the stakeholder functional and non-functional requirements. Authors in [1] introduce FAMILIAR, a textual and executable DSL to facilitate large-scale FM management. Authors in [6] propose a template-based approach for mapping FM to concise representations of variability in different kinds of other model. In [11] the authors proposes an approach based on Feature-architecture mapping for automating the derivation by generating some java sources code using Acceleo and ATLAS Transformation Language.

These works highlights various obstacles we will face when implementing the process: managing a large-scale FM, configuring and deriving the FM based on a specification, transforming the FM to create reusable artefacts. Before that, the following sections summarize the studies that have been conducted to date.

Analysis and feature modeling : The SPL engineering proposes three approaches to analyze commonalities and variability: proactive, reactive, and extractive [3]. In the proactive approach, products are planed first and feature model (FM) are designed before software development. In the reactive approach, analysis and modeling are iteratively performed during the software development. Regarding the extractive approach, SPL re-engineering, the core assets are built from an existing product which is not initially built with a SPL method. Our industrial context, which already implements some interoperability mechanisms, led us to opt for the extractive technique.

For this purpose, we start by listing the existing connectors currently in use.

Industrial Case : In order to identify the variability of the connectors, we conducted a study on the existing data exchange techniques practiced at Berger-Levrault (BL)[1], our industrial partner. Currently, BL operates four types of communication, namely APIs interaction, APIs interaction based on central bus, file transfer and a publish/subscribe middleware API called BL-MOM. The APIs communication allows two APIs to communicate through the network. Services expose APIs that can be accessed by others services. APIs communication through bus is similar to simple APIs interaction. In the second case, the APIs don't communicate directly, but sent requests to an intermediate bus and receive responses from it. For the file transfer, an application produces files containing information and deposit it in a place where other must

[1]Berger-Levrault is a software provider specialized in the fields of education, health, sanitary, social and territorial management.

consume. BL-MOM connectors establishes routes between communicating programs following the AMQP protocol. The exchange of messages is handled according to publish/subscribe or asynchronous request/response patterns, consequently allowing the programs to be loosely coupled. BL-MOM uses RabbitMQ[2], a reliable open source communication mediator, as the underlying message broker and provides helpers to facilitate creating messages schema and connectors (publishers or consumers) with messaging operations over this broker.

Variability modeling of the existing connectors : Thanks to the specification of existing connectors combined with the analysis of their code, we represent the variability of the connectors in a FM which is a tree-like representation. The proposed model is read from top to bottom and can be read freely from left to right or vice versa without priority order. Features are described with labelled boxes and can have several child features. At each level, the feature represents the commonality and is considered a root for its child features which correspond to the variability. There are a number of constraints between the features presented in Figure 2. There are different ways to represent the variability [14]. In this paper, we opted for Feature Oriented Domain Analysis (FODA) [9]. At this stage, the FODA notation is suffucient to represent the variable aspects of existing connectors in order to support the idea that it is possible to consider them as a products line.

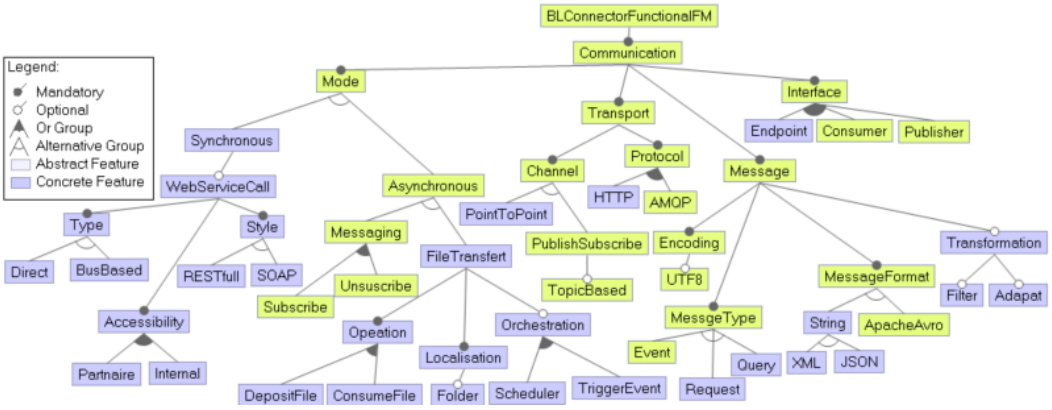

Figure 2: FM representing variability of the existing connectors. Considering the yellow boxes, we obtain an example of FM configured for the BL-MOM connector.

3 Conclusion and Future works

We analyzed existing connectors in service for our industrial partner BL. This first study allowed us to propose a FM based on the analysis of specification and source codes of connectors. The results of the variability analysis supported the idea of considering connectors as a product line. Thus, this paper presented a process for generating interoperability connectors using SPL techniques with the support of the MDE approach. This article also presents some barriers that we will have to tackle to make this approach a reality. Future works will mainly focus on the achievement of the proposed approach. To do this, we will first consider a case study corresponding to a precise type of connector, for example the BL-MOM connector presented in the Figure 2 in yellow color. Thus, we will propose a method to configure a FM in order to

[2] https://www.rabbitmq.com/

obtain the FM of a desired connector. Then, we will work on transforming the FM into reusable artifacts. We will explore the product configuration using other variability modeling techniques such as text variability modeling or domain specific language, which will be compared to the the feature model representation.

References

[1] Mathieu Acher, Philippe Collet, Philippe Lahire, and Robert B France. Familiar: A domain-specific language for large scale management of feature models. *Science of Computer Programming*, 78(6):657–681, 2013.

[2] Mohsen Asadi, Samaneh Soltani, Dragan Gasevic, Marek Hatala, and Ebrahim Bagheri. Toward automated feature model configuration with optimizing non-functional requirements. *Information and Software Technology*, 56(9):1144–1165, 2014.

[3] Noor Hasrina Bakar, Zarinah M Kasirun, and Norsaremah Salleh. Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review. *Journal of Systems and Software*, 106:132–149, 2015.

[4] Nelly Bencomo, Amel Bennaceur, Paul Grace, Gordon Blair, and Valérie Issarny. The role of models@ run. time in supporting on-the-fly interoperability. *Computing*, 95(3):167–190, 2013.

[5] Paul C Clements and Linda M Northrop. Salion, inc.: A software product line case study. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 2002.

[6] Krzysztof Czarnecki and Michał Antkiewicz. Mapping features to models: A template approach based on superimposed variants. In *International conference on generative programming and component engineering*, pages 422–437. Springer, 2005.

[7] Krzysztof Czarnecki, Simon Helsen, and Ulrich Eisenecker. Staged configuration using feature models. In *International conference on software product lines*, pages 266–283. Springer, 2004.

[8] Krzysztof Czarnecki, Kasper Østerbye, and Markus Völter. Generative programming. In *European Conference on Object-Oriented Programming*, pages 15–29. Springer, 2002.

[9] Kyo C Kang, Sholom G Cohen, James A Hess, William E Novak, and A Spencer Peterson. Feature-oriented domain analysis (foda) feasibility study. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 1990.

[10] Ralf Kutsche, Nikola Milanovic, Gregor Bauhoff, Timo Baum, Mario Cartsburg, Daniel Kumpe, and Jürgen Widiker. Bizycle: Model-based interoperability platform for software and data integration. *Proceedings of the MDTPI at ECMDA*, 430, 2008.

[11] Nesrine Lahiani, Djamal Bennouar, et al. A dsl-based approach to product derivation for software product line. *Acta Informatica Pragensia*, 5(2):138–143, 2017.

[12] Kung-Kiu Lau, Perla Velasco Elizondo, and Zheng Wang. Exogenous connectors for software components. In *International Symposium on Component-Based Software Engineering*, pages 90–106. Springer, 2005.

[13] Saleh Majd, Abel Marie-Hélène, Misséri Véronique, Moulin Claude, and Versailles David. Integration of brainstorming platform in a system of information systems. In *Proceedings of the 8th International Conference on Management of Digital EcoSystems*, pages 166–173, 2016.

[14] Raúl Mazo, Camille Salinesi, Daniel Diaz, Olfa Djebbi, and Alberto Lora-Michiels. Constraints: The heart of domain and application engineering in the product lines engineering strategy. *International Journal of Information System Modeling and Design (IJISMD)*, 3(2):33–68, 2012.

[15] Parastoo Mohagheghi and Jan Aagedal. Evaluating quality in model-driven engineering. In *International Workshop on Modeling in Software Engineering (MISE'07: ICSE Workshop 2007)*, pages 6–6. IEEE, 2007.

[16] Lionel Seinturier, Philippe Merle, Romain Rouvoy, Daniel Romero, Valerio Schiavoni, and Jean-Bernard Stefani. A component-based middleware platform for reconfigurable service-oriented architectures. *Software: Practice and Experience*, 42(5):559–583, 2012.