The SAPIENSA approach for service-enabling pre-existing legacy assets

The SAPIENSA Approach for Service-enabling Pre-existing Legacy Assets

Maryam Razavian[1], Dinh Khoa Nguyen[2], Patricia Lago[1], and Willem-Jan van den Heuvel[2]

[1] Department of Computer Science, VU University Amsterdam, The Netherlands
{m.razavian, p.lago}@few.vu.nl
[2] European Research Institute in Service Science (ERISS) Tilburg University, The Netherlands
{D.K.Nguyen, wjheuvel}@uvt.nl

Abstract. Migration of legacy assets to SOA embodies a key software engineering challenge. Existing methodologies mostly focus on development of new services while they provide little to no guidance for transforming services from pre-existing enterprise assets. SAPIENSA, a joint research project of the VU University Amsterdam and Tilburg University, aims to fill this gap. It proposes a migration methodology creating a well-constructed SOA out of pre-existing enterprise assets. The methodology exploits the relevant architectural knowledge to drive migration. To this end, the necessary knowledge concerning the migration should be identified, captured, analyzed and generalized. It should be noted that, the methodology is the result of an extensive analysis of the literature in the fields of reengineering and service engineering. This paper provides an overview of the SAPIENSA methodology along with discussions on the regarding research areas.

1 Introduction

Facilitating the reuse of existing business functions from legacy systems in development of new (service based) systems has become one of the major challenges of modern service engineering methodologies. Software services mostly draw on the functionality of pre-existing enterprise information systems. Some of these may be legacy systems while others may still be technically-healthy and value-adding applications for an enterprise. However, existing service engineering methodologies mostly focus on development of new services while offering very limited support for discovering candidate services from pre-existing software assets.

Since the early use of SOA, migration of legacy assets to services has caught a lot of attention in research and industry. However, it is mostly assumed among researchers and developers that, existing enterprise assets are made to act as services simply by creating wrappers and leaving the underlying implementation untouched. Likewise, enterprises are spending a significant amount of their time assembling applications that provide Web service functionality rather than worrying about the design principles and patterns that guide the development of enterprise services from existing assets. This mainly comes with two disadvantages. Firstly, the architecture of the original application is jeopardized and

as a result the new system delivers lower quality than the pre-existing system. Secondly, due to lack of conformance to service engineering principles, SOA promises, such as promoting highly standardized, loosely coupled services to foster easy composition of distributed applications [1], may not be achieved. To resolve these disadvantages, a comprehensive SOA migration methodology is of critical importance for creating a well-constructed SOA out of pre-existing enterprise assets.

Let us consider the example of migrating a legacy system that stores and manages patient's medical records. First, an understanding of existing system functionality, properties and constraints should be achieved. Such information facilitates extraction of legacy element and then transformation of them into relevant service types For instance, if the knowledge regarding business data and functionalities associated to patient is not extracted a smooth transition to patient data service is not possible. All the knowledge concerning the existing patient information system, the target SOA environment (e.g. types of services) and modernization techniques(e.g. wrapping) affect the migration process in terms of "what" is going to be migrated and "how" the migration is performed.

Despite its simplicity, this example already reflects the relevance of knowledge management in the migration methodology. Knowledge management helps to rationalize the investigation of legacy assets as candidate services, isolate their properties and transform them into meaningful business services while adhering to service relevant aspects and challenges.

SAPIENSA (Service-enAbling Pre-exIsting ENterpriSe Assets), a joint research project of the VU University Amsterdam and Tilburg University, aims to fill this gap. SAPIENSA envisions proposing a methodology for service-enabling pre-existing enterprise assets. The methodology exploits architectural knowledge to effectively carry out the migration on different SOA abstraction levels. This paper provides an overview of SAPIENSA and its research objectives. It is organized as follows. Section 2 provides the background on the three main ingredients of SAPIENSA methodology being service engineering methodology, SOA migration and architectural knowledge management. Section 3 presents the SAPIENSA methodology, while Section 4 gives a holistic representation of a gap analysis technique as an instantiation of the transformation step of SAPIENSA. Finally, Section 5 concludes the paper.

2 Background and Motivation

2.1 Meet-in-the-middle Service Development Strategy

Service engineering methodologies such as [1–3] provide guidelines, models, best practices, standards and reference architectures necessary to construct well-defined SOA. They may follow one or more *service development strategies* for identifying, conceptualizing, designing and implementing services for a new SOA. Typically, conventional approaches support either a top-down or a bottom-up strategy:

Top-down: In this strategy processes are usually deployed in a top-down fashion. The main benefit of the top-down service development is the consistency of the applications and integration mechanisms. It is also rather easy to evolve

the service-oriented solution across the enterprise as the industry evolves. On the downside, this strategy are the costs involved in development as well as the costs of achieving consensus on a high-level SOA architecture throughout the enterprise [4].

Bottom-up: This strategy focuses on development of new services out of existing applications. This strategy usually involves creating a Web service interface from the API of the existing applications. Bottom-up development is well suited for an environment that includes several heterogeneous technologies and platforms or uses rapidly evolving technologies.

However, our observation shows that contemporary widely adopted service engineering methodologies like [1–3] do not strictly follow only top-down or bottom up strategy. Rather, they share the general consensus that top-down and bottom-up have to be mixed in projects in order to be successful, resulting into the adoption of the *Meet-in-the-middle* (or Middle-out) development strategy. This strategy can be used when an already existing Web service interface - for which an implementation exists - is partially mapped onto a new service or process definition. The Meet-in-the-middle development strategy offers a middle ground that attempts to take advantage of some of the benefits of the other approaches while extenuating some of the most notable problems and risks.

Meet-in-the-middle has become a de-facto development strategy suggested in most of all current engineering methodologies, both in the industry [3] and academia [1, 2]. Still, these methodologies provide too general or too little support for the development of new and rehabilitated services e.g. by providing some abstract guidelines and principles. To resolve this shortcoming, SAPIENSA envisions a comprehensive and in-depth meet-in-the-middle service migration methodology.

2.2 SOA Migration: A modernization technique

As mentioned, a key challenge of service design is to rehabilitate pre-existing enterprise assets into modern services that can smoothly operate with novel business processes [1]. According to [5], three main modernization techniques facilitate the rehabilitation of legacy assets: *Redevelopment*, that requires the system to be redeveloped from scratch, *Wrapping* which surrounds existing data, programs, applications and interfaces with new interface and *Migration* that moves the system to a new platform, while retaining the original system data and functionalities.

Redevelopment is generally an expensive and risky task. This is especially the case for legacy systems that should be modernized into service-oriented systems, having critical characteristics such as continuous availability.

Wrapping entails a rather popular approach towards modernization since it is conceptually simple, requiring limited development costs and preserving past investments in pre-existing assets. On the downside, unless legacy elements are "inherently" compliant with service aspects and challenges (e.g. loose coupling, high cohesion, composability), wrapping could come with some serious drawbacks such as decreased quality of the migrated system and/or architectural erosion.

Migration addresses examination and alteration of the existing legacy system to reconstitute it in the target service based system. This modernization technique embraces eliciting the legacy fragments that are suitable for migration to SOA, altering and reshaping the legacy elements to services and renovating the target system based on transformed services as well as new requirements [6].

Among the three modernization techniques, SAPIENSA adopts a migration strategy that is realized by reusable transformation solutions (e.g. transformation patterns) to achieve a repeatable and controlled legacy to SOA migration. A vast body of work in the area of SOA migration mostly focuses on exposing legacy code as (web) services [7]. Typically, the focus of these works is limited to implementation aspects of migration which usually covers techniques to alter a segment of legacy code to web services. A further family of approaches aims for covering the whole migration process. These approaches are comprised of two main sub-processes: top-down service development and bottom-up service extraction [8, 9]. Other approaches such as [10, 11] address SOA migration problem from strategic analysis point of view.

These methods reflect different perspectives on SOA migration. They mainly differ in the way they provide solutions for two main problems of "what" can be migrated (i.e. the legacy elements) and "how" the migration is performed (i.e. the migration process). In addition, these solutions are associated to different levels of abstraction ranging from implementation to conceptual level. The questions that we address in SAPIENSA are: "What are the possible types of migration (regarding different levels of abstraction)?" and "What are the drivers guiding each of them?"

2.3 Architectural Knowledge Management

Recent work in-the-field of SA shows a shift in how software architecture is perceived: from architecture as the structure of an IT solution [12] to architecture as a set of design decisions [13]. From this emerging perspective, management of Architectural Knowledge including architectural design decisions and their rationale (AK) has become a prominent theme in SA research [12]. We define AK as the integrated representation of the SA of a software-intensive system (or a family of systems), the architectural design decisions, and the related rationale [14]. Existing work on architectural knowledge management mainly focuses on capturing and representing in general terms as such it is relevant for any application domain. To date, there are limited results in identifying the architectural knowledge relevant for SOA. A specific type of AK is contained in architectural and design patterns, which document standard solutions to recurring problems in software design. Some research is emerging in identifying SOA patterns [15–17], addressing several important aspects of service development and management. In these approaches, disciplined design decisions and compliance to SOA patterns are recognized as key criteria for successful development. Still, the documented SOA-related knowledge mostly captures reusable technical solutions, while the related rationale or decision process that led to the solution is lost. SAPIENSA we will identify and codify the AK enabling SOA migration. In this way, decision making process for SOA migration will be fully supported without any lost of knowledge.

3 SOA Migration Methodology: Constituent Phases

As mentioned, the ultimate goal of SAPIENSA is to effectively exploit architectural knowledge to incrementally migrate existing software assets into new service-enabled system. To this end, in this section we present our envisioned solution from two different perspectives of methodology and architectural knowledge management.

From methodological point of view, SAPIENSA pursues the following meet-in-the-middle approach by combining:

- In a top-down manner, the end-to-end process portfolio is decomposed into a well-structured and well-designed business service portfolio
- From bottom up, the legacy elements suitable for migration to SOA are extracted and eventually transformed to relevant service types

From knowledge management point of view, SAPIENSA seeks to devise a reference knowledge model capturing types of knowledge that drive SOA migration. To this end, we envision to analyze different migration processes (from industrial cases) and extract the implicit and explicit types of knowledge, which shape the migration. We propose the phases depicted in Figure 1 for the service-enabling methodology using architectural knowledge (AK). As mentioned, this methodology is the result of an extensive analysis of the literature in the fields of reengineering and service engineering. We argue that these phases facilitate extraction of a reference AK model as it helps gathering architectural insights and transforms them to well-constructed services. Figure 1 illustrates the methodology as transformations among artifacts residing in different levels of abstraction. To provide an overview, we generalize the abstraction levels into two categories of conceptual and system level. The conceptual level addresses knowledge models while the system level deals with actual running systems.

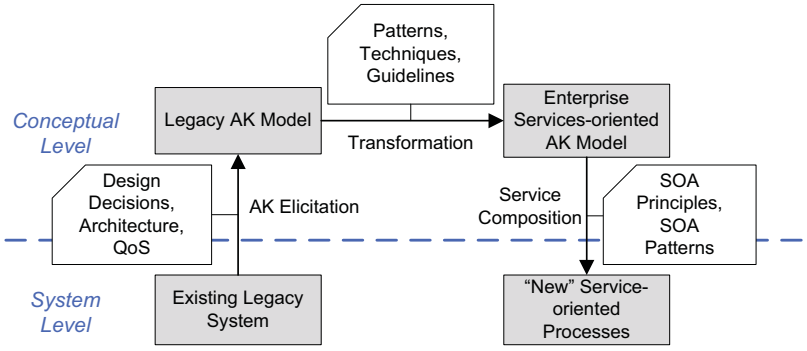

Fig. 1. SAPIENSA Methodology Constituent Phases

3.1 Architectural Knowledge Elicitation

The main goal of this phase is to provide necessary foundation for identification of legacy elements that are relevant for migration to SOA and make the decision regarding how to transfer to services. To this end, we aim at identifying, isolating and describing the knowledge about the typical "standard" elements that belong to pre-existing enterprise assets, and generalize the reusable knowledge. To clarify the idea, we recall the example presented in Section 1. The following types of knowledge could shape the migration process of the medical information system: the body of knowledge concerning the properties of existing system including lost abstractions (structural design, business processes, business rules, etc.), concerns (i.e. quality attributes such as performance that are desired to be found back in the migrated service based system) and know-how (design decisions, discarded alternatives, etc.). we argue that a reference knowledge model capturing types of knowledge that shape the migration, provides the basis for migration decision making process. At this point, the following research question arises.

- *RQ1) what are the knowledge elements that are relevant to be made explicit?* For instance, solution-related knowledge such as architecture, design decisions, etc., as well as problem-related knowledge including functional requirements (e.g. business processes and business rules) and non-functional requirements (e.g. privacy and security), should be externalized.

In order to enable an effective knowledge capturing and sharing, the knowledge extracted in this phase should be explicitly represented. Architectural knowledge models aiming at representing decisions in general are considered as a potential means of codifying the elicited knowledge. For instance, the AK core model developed in our previous work [18] is relevant for making AK explicit. Knowledge elements specific to an organization or an application domain should be defined as specific extensions to the core model described above. We consider service-oriented enterprise architectures as SOA extensions. From a methodological point of view, this phase entails the process of analyzing the existing system to identify the system's structure, functionality and behavior and identifying the elements constituting candidate services.

3.2 Transformation

The transformation phase embraces the actual migration of legacy assets to service based assets. The AK acquired during the AK elicitation phase will be used as a fabric for creating new business services. This transformation could be in the form of reshaping design elements, restructuring the architecture and/or altering business models and business strategies. It should be noted that, each one of these forms of transformation belongs to a specific level of abstraction. For instance, a legacy element can be transformed to services by altering its encapsulation using wrapping techniques (system level transformation). In the same way, a composition of legacy elements can be transformed to a service or composition of them. At a higher level, an existing business model is transformed to a to-be business model based on new requirements as well as opportunities offered

by service based systems (conceptual level). In all these cases the knowledge element extracted within AK elicitation phase is converted to another knowledge element in the SOA environment. Feasibility of each type of migration depends on many additional factors, such as past design decisions and knowledge about extra functional aspects like technological constraints or quality attributes. It becomes therefore even more important to capture the relevant AK in pre-existing and new systems in order to assess the feasibility of different transformation techniques. Recall that in this research work, we aim at isolating and describing the AK about the typical "standard" elements that belong to pre-existing enterprise assets; in the same way, we want to describe the AK about the "standard" elements of an SOA, including the architectural patterns relevant for SOA transformations like for instance service topologies and worst-case quality concerns. By carrying out a number of industrial case studies, we will define how we can typically transform a standard element of preexisting enterprise assets, into a "standard" element of an SOA. This will result in a number of transformations, as represented in the horizontal, continuous arrow in Figure 1. In this way, the "SOA migration decision making process" is supported by a knowledge base of SOA transformations capturing necessary technical and nontechnical AK. In order to address all these aspects of transformation, following research questions should be answered:

- *RQ2) what are the elements which drive the transformations?* For instance, past design decisions, quality attributes,technical constraints or business artifacts can have influence on the solution regarding the transformation problem.
- *RQ3) what are the possible types of transformation regarding different levels of abstraction?*
- *RQ4) how to represent the body of knowledge associated to transformation?*

From methodological point of view, the transformation phase focuses on "how" existing legacy elements are transformed into services. This process is incremental and interactive. The transformation phase results in set of enterprise services (business and infrastructure) which encapsulate a legacy elements.

3.3 Service Composition

Typically, end-to-end service-based business processes are artifacts reflecting the functional and non-functional business requirements of an enterprise in end-to-end value chains. These processes need to be designed in terms of constellations of several interacting services obtained from the transformation phase. This is supported in the service composition phase, which provides the ability to restructure, compose and decompose services by means of Service Composition Model (SCM). In particular, leveraging the SCM, the designers should be able to compose (or decompose) and relate services that are obtained from pre-existing enterprise systems, developed from scratch, or leased from external service providers. Besides SCM should reflect bodies of knowledge which shape the service compositions. For instance, SOA principles such as cohesion, low communication, autonomy and loose coupling play a key role in the service composition phase. In addition, external business and design constraints such

as service associations, message exchange behaviors (i.e. business protocols) and other non-functional requirements (e.g., Quality of Service (QoS), security, resource and platform constraints) have to be considered while (re-)composing the services. To sum up, following research questions are raised:

- *RQ5) What are the relevant knowledge elements (e.g. SOA design principles, constraints imposed by business domain or target service based system) that drive the service (re-)compositions and how they should be fulfilled?*
- *RQ6) How do the existing constraints of the pre-existing systems affect the service compositions?* For instance, existing architectural, system, and resource constraints (e.g. modularity levels, system performance, throughput) that are desired to be found back in the new service based system, should drive service compositions.

4 Gap Analysis for Service Transformation

In order to exemplify the transformation phase, in this section, we present our work on model driven gap analysis approach [19, 20] as a transformation at conceptual level. Basically, this approach detects and assesses the reuse of available software assets for implementing (parts of) the newly conceived business services that collectively shape the novel service-oriented business processes.

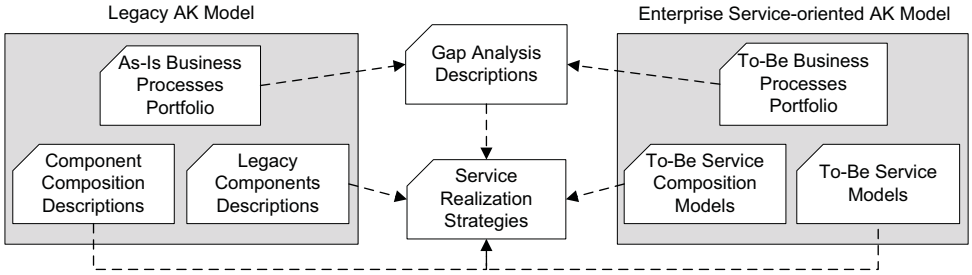

Fig. 2. Gap Analysis for Service Transformation

Figure 2 presents the positioning of the Gap Analysis on the transformation phase (SAPIENSA step 2). Inputs of the Gap Analysis include the abstract as-is and to-be business process portfolio. The as-is business process portfolio is inherently architectural knowledge recovered from pre-existing (internal and external) software assets (*legacy AK models*) during the first phase of SAPIENSA. These knowledge elements allow the various stakeholders to understand the portfolio of available (internal and external) applications and business processes. In contrast, the to-be business process portfolio contains the desired novel business processes introduced to address current and future enterprise needs. They are categorized as *Enterprise Service-oriented AK models* that address the target SOA environment and could be elicited from various sources such as industrial

best practices, customized reference models, or companies' own business process innovations and improvements. The results of Gap Analysis represents the commonalities and discrepancies between the as-is and to-be process models and help to identify the appropriate realization strategies for the services and service compositions. Selecting appropriate realization strategies necessitates also the architectural knowledge on the lower abstraction levels, i.e. component and composition models of both the existing legacy systems and the new desired service-oriented systems. A resulting service realization strategy includes design decisions considering whether (parts of) the new service functionalities can be obtained by reusing or revamping existing IT assets, leveraging external IT services, or whether they should be re-designed or developed from scratch. Service realization strategies serve as the input AK for the lower-level service transformation, e.g. transforming a atomic software component into an atomic Web Service or a composition of components into a composite service.

5 Conclusion

Existing service engineering approaches provide little to no guidance to migrate pre-existing legacy systems to SOA. To resolve this shortcoming, a methodology-based approach supporting extraction of legacy assets as candidate services as well as decomposition of business processes and services down to the level that could be associated to these legacy assets is desired. As discussed, the SAPIENSA project envisions a comprehensive, knowledge-driven meet-in-the-middle service migration methodology for developing SOA-based applications out of pre-existing enterprise assets.

This paper has presented the motivations and research objectives of the SAPIENSA project. In addition, a framework representing a holistic representation of our proposed SOA migration methodology including three constituent phases (AK elicitation, transformation and service composition) along with their research challenges was presented. In order to better clarify the transformation step, that is the key activity of SOA migration, the gap analysis method has been introduced as a type of the transformation. Besides, the mapping of gap analysis on the SAPIENSA framework facilitates to better characterize its properties in terms of artifacts included, types of knowledge exploited, and the dependencies among them.

The results of SAPIENSA is preliminary in nature. The next step is to find out how migration process is performed in industrial practice, in terms of both activities carried out and types of knowledge exploited. We employ the action research approach including the following cycle. After defining the methodology to be used and the AK reference model, a number of iterations will occur. In each iteration, the cases carried out will be followed by a reflection period aimed at giving feedback on the results. Lastly, all results will be consolidated in a generalized and reusable solution.

6 Acknowledgment

This research has been partially sponsored by the Dutch Joint Academic and Commercial Quality Research and Development (Jacquard) program on Software Engineering

Research via contract 638.001.206 SAPIENSA: Service-enAbling PreexIsting ENterprISe Assets; and the European Community's Seventh Programme FP7/2007-2013 under grant agreement 215483 (S-Cube).

References

1. Papazoglou, M.P., Heuvel, W.J.: Service oriented architectures: approaches, technologies and research issues. The VLDB Journal **16**(3) (2007) 389–415
2. Zimmermann, O.: An Architectural Decision Modelling Framework for Service-Oriented Architecture Design. PhD thesis, Department of Computer Science, Stuttgart University (2009)
3. Arsanjani, A., Ghosh, S., Allam, A., Abdollah, T., Ganapathy, S., Holley, K.: SOMA: A method for developing service-oriented solutions. IBM System Journal, Vol 47., No.3 (2008)
4. Graham, S., Simeonov, S., Boubez, T., Daniels, G., Davis, D., Nakamura, Y., Neyama, R.: Building Web Services with Java. 2nd edn. SAMS Publishing (2005)
5. Bisbal, J., Lawless, D., Wu, B., Grimson, J.: Legacy information systems: Issues and directions. IEEE Software **16** (1999) 103–111
6. Razavian, M., Lago, P.: Towards a conceptual framework for legacy to soa migration. In: Presented in Fifth International Workshop on Engineering Service-Oriented Applications (WESOA'09). (2010)
7. Sneed, H.M.: Integrating legacy software into a service oriented architecture. In: CSMR '06: Proceedings of the Conference on Software Maintenance and Reengineering, Washington, DC, USA, IEEE Computer Society (2006) 3–14
8. Zhang, Z., Yang, H.: Incubating services in legacy systems for architectural migration. In: APSEC '04: Proceedings of the 11th Asia-Pacific Software Engineering Conference, Washington, DC, USA, IEEE Computer Society (2004) 196–203
9. Andreas Winter, J.Z.: Model-based migration to service-oriented architecture. In: the Int. Workshop on SOA Maintenance Evolution (SOAM 2007). (2007)
10. Lewis, G., Morris, E., Smith, D., O'Brien, L.: Service-oriented migration and reuse technique (smart). In: STEP '05: Proc of the 13th IEEE International Workshop on Software Technology and Engineering Practice, Washington, DC, USA, IEEE Computer Society (2005) 222–229
11. Umar, A., Zordan, A.: Reengineering for service oriented architectures: A strategic decision model for integration versus migration. Journal of Systems and Software **82**(3) (2009) 448 – 462
12. Shaw, M., Clements, P.: The golden age of software architecture. IEEE Softw. **23**(2) (2006) 31–39
13. Bosch, J.: Software architecture: The next step. In: European Workshop on Software Architectures (EWSA), Springer Verlag (2004) 194–199
14. Muhammad Ali Babar, Torgeir Dingsyr, P.L.H.v.V.: Software Architecture Knowledge Management Theory and Practice. Springer (2009)
15. Erl, T.: SOA design patterns. Prentice Hall (2009)
16. Zimmermann, O., Grundler, J., Tai, S., Leymann, F.: Architectural decisions and patterns for transactional workflows in SOA. (2007) 81–93
17. Zdun, U., Hentrich, C., Dustdar, S.: Modeling process-driven and service-oriented architectures using patterns and pattern primitives. ACM Trans. Web **1**(3) (2007) 14
18. de Boer, R.C., Farenhorst, R., Lago, P., van Vliet, H., Clerc, V., Jansen, A.: Architectural knowledge: Getting to the core. In: QoSA. (2007) 197–214
19. Nguyen, D., van den Heuvel, W., Papazoglou, M., de Castro, V., Marcos, E.: Gap analysis methodology for business service engineering. In: Proc. of the 11th IEEE Int. Conference on Commerce and Enterprise Computing (CEC09). (2009)
20. Nguyen, D., van den Heuvel, W., Papazoglou, M., de Castro, V., Marcos, E.: GAMBUSE: A Gap Analysis Methodology for Engineering SOA-based Applications. In: Conceptual Modeling – Foundations and Applications. Springer-Verlag (2009)