Towards Systematic Management of Third-party Services used within Long-living Business Information Systems

Christof Momm, Frank Schulz
SAP AG Research Center CEC Karlsruhe
Vincenz-Priessnitz-Strasse 1, 76131 Karlsruhe, Germany
Email: {christof.momm | frank.schulz}@sap.com

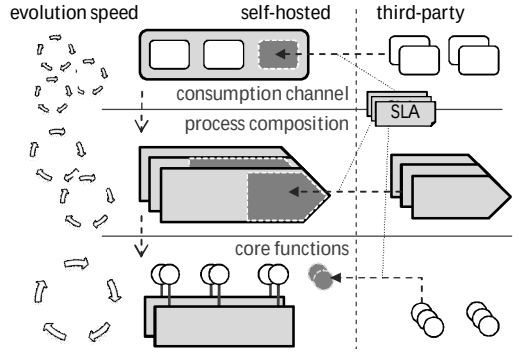
1 Introduction

Today, companies must constantly adapt to changing market requirements. Thus, long-living business information systems (BIS) used for supporting business processes have to offer the flexibility to evolve according to these changing requirements. Otherwise, companies have to cope with the problem presently well known under the term "legacy systems".

In this paper, we propose a general architecture blueprint for long-living BIS helping to avoid the legacy problem and to protect the huge investments made in BIS. One central feature of this architecture is that the integration of innovative third-party is particularly well supported. The second part of this paper therefore addresses the problem of managing external third-party services based on service level agreements (SLA) and introduces a systematic management approach.

2 Timeless Business Information Systems

The following figure introduces an architecture blueprint for "timeless" business information systems, supporting continuous adaptation to changing requirements and technological innovations on top of a fairly stable core.



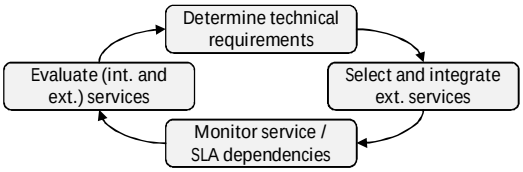
The required flexibility is basically supported by a layered service-oriented architecture, e.g. according to [1]. During evolution of long-living BIS, these different layers will evolve at different speed. Thus, evolution speed is also a criterion for identifying services on the different layers. The lowest layer comprises components and systems providing a set of core functions required for running the business. These functions are exposed in terms of interoperable application services, e.g. basic services for processing sales orders. We expect the fundamental requirements for such core business functions to be fairly stable, resulting in rather long evolution cycles. To support flexibility in case of changes in the market, the architecture introduces the process composition layer, which therefore possesses a significantly higher evolution rate. Here, workflow management techniques are leveraged for implementing custom processes on top of the set of core services, e.g. a custom sales order process, which has to be constantly adapted to changing partners or the internal work organization. However, the highest evolution speed is expected on the layer of end user consumption channels, such as mobile devices, web-based applications, mash-ups etc. This is mainly due to the very short cycles of technological innovations in this area and their significant impact on the market expectations. Customers demand for early adoption of new channels, e.g. smart phones.

Besides this kind of decoupling, another key to timelessness, in our opinion, is to explicitly support the integration of external third-party services. On each layer, there should always be the choice between self-hosting the services on premise and "importing" services from a third-party provider. If so, SLAs are used to compensate for the lack of control over the quality of external services. Regarding the core components/systems, we expect most of them to be hosted on premise, in particular business critical functionality operating on sensitive data. By contrast, process compositions and channel components (e.g. ready-to-use remote portlets or mash-ups) are particularly well-suited for being developed and provided by third-party providers. However, business-critical components are probably still hosted on premise.

3 Management of Third-party Services

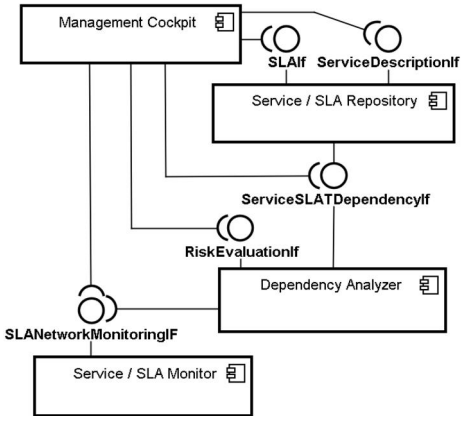
The integration of external third-party services in the BIS based on SLAs requires a systematic approach and adequate tool support. In the following we introduce the major activities when

dealing with external third-party services and point out unique selling points for our proposed solution.



First, the internal provider has to determine quality requirements for external services based on internal objectives. Afterwards, the provider has to select external services meeting these requirements. Whereas most related approaches limit the scope to translating requirements and negotiating services, we target a more extensive decision support for the internal provider, including an economic risk evaluation of different service and service provider combinations. This requires a very specific service/SLA monitoring architecture supporting the monitoring of dependencies between internal and external services/SLAs. Unlike many existing approaches, we also focus on supporting a consumer-centric monitoring of the third-party services, which complements rather coarse-grained and less trustworthy SLA reports available from external providers. The collected information forms the basis for a continuous evaluation of existing service combinations and quality dependencies as starting point for refining technical requirements and improving the selection of third-party services.

To support such a closed-loop management of third-party services we developed the following framework architecture:



The Service Repository represents the central data store for managing external and internal services as well as relationships between them. It includes service descriptions, corresponding SLA templates and agreed SLAs using existing models, in particular [2], as well as custom models for capturing technical dependencies between services and SLA templates. The technical dependencies are determined using a statistical testing approach based on [3] and are later on used for selecting the required external services. As soon as external services are selected, integrated and actually used, the Dependency Analyzer component continuously evaluates economic risk for different combinations of external and internal services using portfolio theory [4]. This risk evaluation based on historic information complements the technical dependency information during service selection. To interact with the framework the service provider generally uses the Management Cockpit. This graphical user interface offers access to all supported management functions. Additionally, it visualizes the current state of the SLA and service landscape for manual analysis of complex dependencies.

4 Current Status and Future Work

So far, we have developed an ad hoc prototype of the framework already offering basic support for all activities pointed out before. To evaluate the framework we integrated it with a demonstrator that follows the timeless BIS architecture blueprint. This demonstrator comprises SAP Enterprise Services, which in combination with third-party optimization services form the stable set of core functions. Based on these services, we created a custom production planning process using Netweaver Composition Environment. This process is accessible via a web-based portal channel. Decision support functionality focuses on selecting optimal combinations of Enterprise Services with different external optimization services.

In future, we will intensify research on translation of quality requirements in order to support a larger variety of service types and quality metrics. In addition to this, we will further elaborate the risk evaluation algorithms for combinations of services.

Acknowledgement: This work is supported by the German Federal Ministry of Education and Research under promotional reference 01|G09004 (ValueGrids).

References

[1] Papazoglou, M: Web Services: Principles and Technology, Addison-Wesley, 2008

[2] Andrieux, A./K. Czajkowski et al.: Web Services Agreement Specification (WS-Agreement), 2007

[3] Happe. J./D. Westermann et al.: Statistical Inference of Software Performance Models for Parametric Performance Completions, 6th International Conference on the Quality of Software Architectures (QoSA 2010), 2010

[4] Michalk, W./B. Blau et al.: Risk-Based Decision Support in Service Value Networks, 43rd Annual Hawaii International Conference on System Sciences (HICSS 2010), 2010