Conceptual Modelling of Service-Oriented Systems

Mario A. Bochicchio[1], Vincenzo D'Andrea[2],
Natallia Kokash[2], and Federica Longo[1]

[1] SetLab, University of Salento, Italy {mario.bochicchio, federica.longo}@unile.it
[2] DIT - University of Trento, Italy {natallia.kokash, vincenzo.dandrea}@dit.unitn.it

Abstract. The design of service-oriented systems currently is one of the most important issues in the software engineering. In this paper, a conceptual framework for designing Web service-based systems is presented. This approach is characterized by client-centered analysis and presence of business-process modelling to identify functionalities and collaboration patterns of involved Web services. Service discovery and selection are parts of the design process. A case study is provided to explain the principle steps of the proposed framework.

1 Introduction

Web services are self-contained software units that can be published, located and invoked via the Internet. They are seen as the future of the Web simplifying business integration and achieving a separation of concerns. Many organizations are interested in wide acceptance of Service-Oriented Architectures (SOAs) and Web service standards, leveraging both from a perspective to use the Web as a market for their own services and ability to consume already existing ones. In practice, this technology brings multiple technical and conceptual problems. One of them concerns the development of Web services and their further reuse in more complex systems. It is assumed that designers of such systems either are aware about existing Web services and can model systems able to collaborate with them or describe abstract services they would like to use relying then on automated (even runtime) service discovery to substitute these abstract descriptions with real Web services. None of these scenarios is really acceptable. In the former case, all information about a required Web service should be known in advance (before starting actual system design) to allow for its location in a UDDI registry or a like. In the latter one, it is unlikely that exactly such a service will be found and, therefore, a bulk of problems with service or system adaptation to achieve interoperability appear. To resolve this issue, a design methodology that follows the principle "meet-in-the-middle" is required.

Another question is what functionalities should be implemented as Web services to be both self-contained and allow for their reuse in different contexts. Who are their intended clients? Such functionalities can be singled out through analysis of existing business processes to accomplish logically independent tasks that are not highly specific for a given process. It means that while modelling a system designers can stand out potentially reusable parts as Web services with additional efforts to make them customizable and acceptable by all members of an intended audience.

The objective of this paper is to find a conceptual design methodology that addresses the above problems. Conceptual modelling of business-level collaboration protocols is essential for attaining the system composed from a set of communicating services, easily replaceable with other services and potentially reusable in different application contexts. The design process in this case requires analysis of all stakeholders of the system and its goals at the abstract level. We adopt the xBPEM (Business Process Enhanced Model Extended with Systemic Variables) [1] framework and extend it for service-capable modelling.

The rest of the paper is organized as follows. In section 2, a necessary background on Web services and service-oriented design is given. Section 3 presents the proposed methodology for modelling service-based business processes. In Section 4, a case study is given. Section 5 summarizes the contributions of this paper and outlines future work.

2 Background

In this section we analyze existing service-oriented modelling techniques.

Web Service Description Language (WSDL) describes a Web service interface by means of operation signatures. Different languages have been proposed to supplement static service interface descriptions with dynamic information about service functionalities. Business Process Execution Language for Web Services (BPEL4WS) includes means for specifying abstract and executable processes. Global behavior of all parties involved in a collaboration can be specified with Web Service Choreography Definition Language (WS-CDL). Use of WSDL, BPEL4WS and WS-CDL for modelling service-based business processes brings two problems: (1) the level of abstraction is too low for convenient modelling, and (2) the languages are lacking a standardized graphical representation which would ease the design process.

Basic design principles of service-based applications have been described by Papazoglou and Yang [2]. A weak point of this work is that it does not distinguish logical business processes from their implementation. Its first step is to "identify, group and describe activities that together implement a business process," which is a sort of bottom-up approach. The presented framework is fully service-oriented, i.e., all activities are modelled as Web service invocations, while in real-world systems other functionalities are needed. Model Driven Architecture (MDA) aims at simplifying business and technology change by separating business and application logics from underlying platform technology. It clearly separates *business processes*, designed based on pure business concepts, from *technical processes*, or software realizations of abstract business processes. Quartel et al. [3] propose Interaction Systems Design Language (ISDL) for graphical modelling of service-oriented systems, considering specific concepts such as internal and external activities.

We suppose that a step from proprietary notations towards Unified Modelling Language (UML), standard, well-known and close to software system design language, should be done. Such approaches take advantage of model-based analysis of semantical interoperability between different components of a system. UML-based service discovery tools (e.g., [4]) can simplify discovery and adaptation of external Web services minimizing risk of logical flaws through analysis of their structure and internal behavior. Another argument towards UML is that Web services represent a new dimension in development of Web Information Systems (WIS). Such systems now can be constructed by means of transparent integration of services available on the Web and graphical Web interfaces. A language for modelling Web services should be compatible with traditional software design techniques that normally employ UML.

Several works study modelling of Web service structural and behavioral aspects using UML. Gardner [5] explains how UML activity diagrams can be mapped into BPEL4WS. Deubler et al. [6] introduce aspect-oriented modelling techniques for UML sequence diagrams. These techniques are needed to specify certain behavior aspects of overlapping Web services (so called *crosscutting services* or *aspects*). An approach by Kramler et al. [7] model Web service collaboration protocols taking into account their specific requirements and supporting mapping of design views into existing Web service technologies. Feuerlicht and Meetsathis [8] define domain-specific service interfaces from a programmatic viewpoint. Lau and Mylopoulos [9] apply Tropos [10] for designing Web services. Tropos is an agent-oriented software development methodology operating with concepts of *actors*, *goals*, *plans*, *resources*, *dependencies*, *capabilities* and *beliefs*. Kazhamiakin et al. [11] use a Tropos-based methodology for modelling business requirements for service-oriented systems, starting from strategic goals and constraints. Penserini et al. [12] propose a Tropos extension for designing services, which allows for an explicit correlation of stakeholder's needs with systems plans and environmental constraints. Aiello and Giorgini [13] show how Tropos can be used to model Quality of Service (QoS) requirements.

There is a need for describing the procedure on how to derive "good" service abstractions from high-level business requirements and business process models (e.g., identify candidate services within a given UML analysis model) [14]. Too coarse grained services tend to have a low reuse potential, while too fine-grained services may not be loosely coupled and require a big amount of message exchange and coordination efforts. An idea of using a meet-in-the-middle modelling approach as opposed to a top-down or bottom-up solutions seems to be effective in this context [15]. A service-based systems must be designed using a combination of business-driven service identification coupled with service identification through leveraging legacy assets and systems.

Design of service-oriented systems that implement abstract cross-organizational business processes is not an easy task. Such systems should lead to creation of reusable context-independent services and be oriented on adaptation of existing

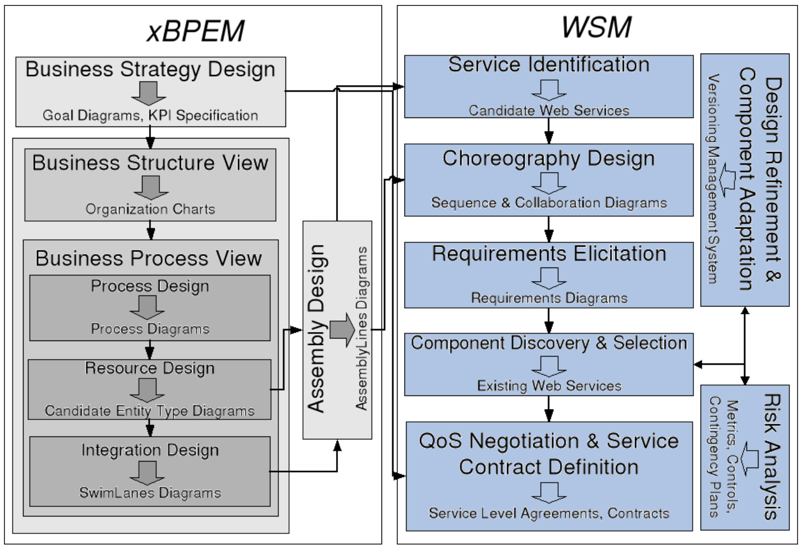


Fig. 1. xBPEM and Web Service Modelling (WSM) frameworks.

legacy components and provision of appropriate interfaces and QoS for all kinds of end users. The goal of the current work is to come with a set of design steps that can help to make good SOA-related decisions. We examine the possibility to adopt xBPEM methodology [1] for designing service-oriented systems.

3 Methodology

In this section we propose our framework for conceptual modelling of service-based business processes.

UML is a standard general purpose modelling language, but it is mainly system oriented. BPEM [16] is an extension of UML for business process modelling. BPEM qualitatively represents business goals in natural or semiformal language but does not consider impact of Web technology on them. xBPEM approach further elaborates BPEM by introducing a consistent business strategy design methodology. This methodology is used to control business goals through business process Key Performance Indicators (KPIs), which enable measurement of both internal process performance and QoS from customer viewpoints.

Figure 1 draws the main conceptual steps of the xBPEM and its extension to enable seamless involvement of Web services into the process. In a nutshell, the proposed framework consists from the following steps:

1. *Business Process Design.* The overall process begins with the business process modelling phase according to the xBPEM methodology. Thus, stakeholders are defined with their high-level goals and basic KPIs, which measure the global performance from the viewpoints of each stakeholder.

2. *Service Identification.* This step is focused on separating loosely coupled functional parts of the collaborative business process into standalone service components. Considering that business processes evolve in time in order to adapt to business strategy which changes following market rules (i.e. new products, new services, new business models), we may observe that candidate services can be identified as:
   - repeated blocks of activities inside a business process;
   - similar blocks of activities among various business processes or different applications;
   - time-invariant blocks of activities in a time-variant business process.

   An accurate analysis of the activity diagrams produced at the first step is essential to identify these blocks of activities and to make them eligible to become Web services. The analysis process can be partially automated by searching for similar graphical patterns in the activity diagrams representing abstract business processes.
3. *Choreography Design.* Desired communication patterns among identified abstract services are defined using UML sequence or collaboration diagrams based on the interaction patterns among stakeholders, Web services and the system which are shown into the xBPEM's swimlanes diagrams.
4. *Requirements Elicitation.* Requirements for discovery of existing software components and Web services (inputs/outputs, pre-conditions/effects, behavior patters and desired quality levels) are defined based on the identified abstract services.
5. *Components Discovery and Selection.* Software components that can be used to compose the system are of two essential types:
   - *Modifiable*, i.e. available as documented source-code (e.g., open source projects) or coming from organizations available to customize it (e.g., software houses);
   - *Unmodifiable*, i.e. coming from providers not particularly interested in customization or not customizable at all (e.g., Web services).

   In particular, this step covers discovery of Web services (interfaces, behavioral specifications and coordination patterns specified at the previous stages). Then, the QoS information about Web services and their providers must be collected (using data published by service providers, authorized agencies or other service clients), in order to select the best candidate Web services.
6. *Risk Analysis.* This step is needed to assess risk related to use of external Web services (loss of service, loss of data, security/privacy concerns, etc.). On the basis of this assessment, the existing risks can be mitigated through selection of more appropriate services, use of alternative services for critical tasks, system re-design, data replication, and so on.
7. *Design Refinement and Component Adaptation.* At this step the prepared models can be refined to allow for the seamless integration of the external Web services. Found Web services and existing legacy systems are tested and analyzed in order to decide how to introduce them to the system. It may happen that adaptors or wrappers are needed. Providers of chosen services

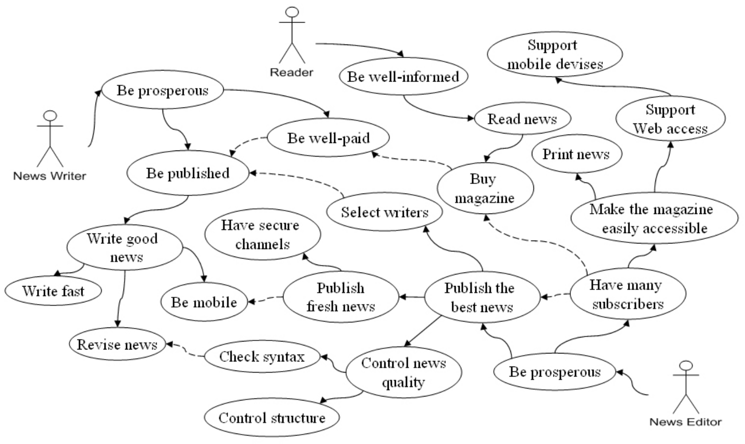


Fig. 2. High-level goal diagram of the news distribution scenario.

become stakeholders of the system. If no services with required functionalities are found, the organization should think about implementing them, thus, increasing code reusability both in its own future projects and in the projects of third parties.

8. *QoS Negotiation and Service Contraction.* If quality parameters of discovered Web services do not correspond to identified KPIs, they can be negotiated with service providers. At this step identified KPIs should be mapped into direct requirements for QoS further resulting in SLAs.

4 Case Study

In this section we consider a case study extracted from a news publishing domain.

The case-study scenario is as follows: *A journalist writes an article via mobile phone or via PC and sends it to the editorial office. The system must identify the author and notify the editor about a news event. In case of successful authorization, the news item is stored in a system archive. The editorial staff can see the registered articles, send comments to the author, ask for news revision, select one or more publication channels, and publish the news. The author is notified that his/her article has been published. To access the news, a reader selects a preferred news channel.*

According to the methodology presented in Section 4 this scenario is modelled with the help of goal diagrams and business process diagrams (Activity 1). For each stakeholder main goals are shown in Fig. 2. Related goals are connected with dotted lines. Then, KPIs are defined from perspective of each user type. Table 1 describes examples of KPIs for news editor and news writer.

Figure 4 shows the news distribution process as the flow of three sub-processes: news writing, news publishing and news reading. Activity diagrams and swimlanes, which describe what each of the stakeholders does within the process,

Table 1. KPIs from the news writer's and news editor's perspectives

Stakeholder	**Indicator**	**Measure**
News Writer	Level of automatization	Number of steps supported by a computer system divided by a total number of steps
	Simplicity of the interface	Number of operations to complete the task
	Time to accomplish a task	Measured in minutes
	Information sharing cost	Time from a data entry to a delivery output
News Editor	Task error prevention	Number of incorrectly published news (i.e., number of news in a wrong section)
	Misprint prevention	Number of syntactic errors in published material
	Integration	Number of data transmission errors
	Security	Number of security problems per annum

are shown. Blocks of activities emphasized with color can be provided by Web services. Based on this diagram the following Web services are identified (Activity 2):

- A service (services) responsible for user authentication via SMS, MMS or the Web. Authentication is a functionality with a constant graphical pattern used in three sub-processes, namely news writing, news publishing and news reading, and potentially can be required in other places of the system or other business processes of this organization.
- A service (services) responsible for interaction between a pair of stakeholders or a pair system-stakeholder via SMS, MMS or the Web. Interaction activity has a constant pattern used in different points of the business process;
- A service (services) responsible for syntax checking. Syntax checking is a time-invariant functionality.
- A service (services) responsible for a payment procedure. Payment is a time-invariant functionality that can be used in other business processes (e.g., for author rewarding in the context of the current application).

At the choreography design step (Activity 3), interaction patterns among stakeholders, Web services and the system are modelled with collaboration diagrams. In Fig. 4 news writer and news editor identification processes are shown. This step helps to elicit requirements (e.g., operation signatures, capabilities, preconditions and effects) for Web services to be discovered or created (Activity 4). So, in our case study, service operations for user identification via SMS/MMS and the Web must deal with a login and a password or sender's phone numbers as input. The output message should contain acknowledgement or rejection, and a reason in the latter case. Moreover, a service must support assignment of *roles* to users. A message must be passed through system controller to enable the permitted functions for this user (e.g., submit news or edit saved news).

The next step (Activity 5) is to find Web services with approximate behavior defined at the previous steps. For example, a set of Web services capable to perform on-line payments and send messages via SMS and email have been discovered in the xMethods service registry[3](see Table 2).

[3] XMethods web service registry - http://xMethods.com/

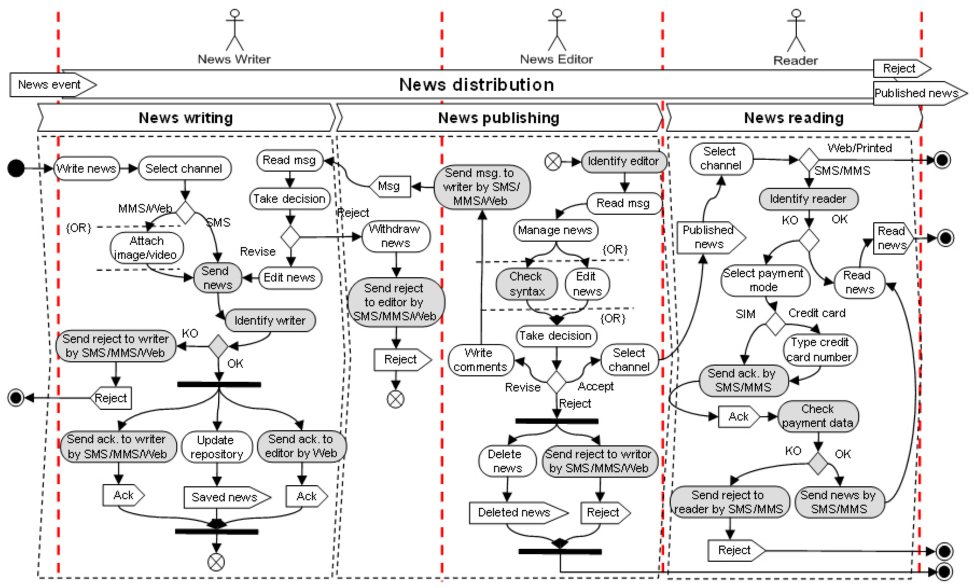
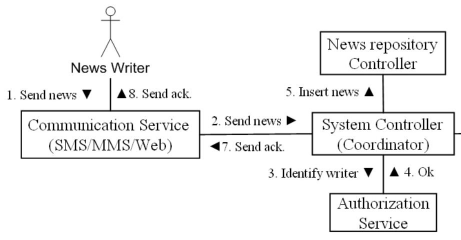


Fig. 3. Activity and swimlanes diagrams for news distribution process.

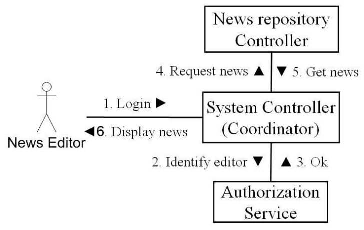
Risk analysis both at technical and business levels is required to establish potential threats affecting the system and damaging stakeholders (through monetary loss, breach of reputation, etc.) at the early stages of the design process (Activity 6). It can be noticed that in our scenario some mechanism is required to collect feedback from a news reader to reveal inappropriate behavior of the payment Web service, unavailability of the syntax checker can be tolerated while a failure of the interaction services may cause severe consequences for the system and, therefore, must be prevented through selection of strategic business partners or service replication.

Thus, after detailed analysis, testing and opportunity evaluation from a business strategy point of view these services can substitute the corresponding activities in our scenario while a Web service supporting interaction via MMS, syntax checker and authorization service should be designed from scratch (or discovered among software components or open source projects). The collaborative diagrams can be refined (Activity 7) to allow for the system's interoperability with the found components. At this step, structure and behavior of existing Web services can be represented by means of UML [17] and included into our model.

Once appropriate Web services have been found, their QoS levels must be evaluated by considering the defined KPIs (Activity 8). Thus, the *security* indicator of news publisher (see Table 1) means that the interaction Web services have to assure secure connection. The *information sharing cost* indicator means, in particular, that the total response time of authorization and interaction Web services and system components involved in news submission process must not



(a) News Writer identification



(b) News Editor identification

Fig. 4. Collaboration diagrams

Table 2. Examples of Web services discovered for the news distribution process

Provider	**Web Service**	**Short description**
richsolutions.com	SmartPayments	Payment Web service that supports credit cards, debit cards and check services
rpamplona	ACHWORKS SOAP (T$$ - Rico Pamplona)	Web services for ACH Processing and Payments (can accept and process check and credit card payments electronically)
adambird	Esendex Send SMS	Sends an SMS message to a mobile phone
StrikeIron	StrikeIron Mobile Email Messaging	Allows SMS to be sent to mobile telephones programmatically for many different service providers
jmf	SendEmail	Send mail messages to any email address

exceed some reasonable value (e.g., 2 minutes) to deal with urgent news like intermediate scores in football matches or preliminary voting results. If a Web service provider does not guarantee high QoS levels by default, they can be negotiated.

5 Conclusion and Future Work

In this paper we have proposed a methodology for conceptual modelling of Web service-based systems following a business modelling approach. In detail, activity diagrams are useful at Requirements Elicitation stage because abstract services are identified through repeated, similar for different processes and time-invariant blocks of the activities which compose a business process. The choreography design among identified abstract services is based on the interaction patterns among stakeholders, Web services and the system described into the swimlanes diagrams. Moreover, the KPIs based on stakeholders' goals are useful for defining desired quality levels for each abstract service at Service Identification stage and for evaluating and trading quality parameters of discovered Web services at QoS Negotiation and Service Contraction stage.

Our future work includes further elaboration and verification of the proposed methodology. In particular, tighter connection with Web service specification formats must be established to enable automated support of the design process for WIS.

References

1. Longo, A.: Conceptual Modelling of Business Processes in Web Applications Design. Phd thesis, University of Lecce, Innovation Engineering Department (2004)
2. Papazoglou, M., Yang, J.: Design methodology for web services and business processes. In: Proceedings of the VLDB-TES Workshop, Springer (2002) 54–64
3. Quartel, D., Dijkman, R., van Sinderen, M.: Methodological support for service-oriented design with ISDL. In: Proceedings of the Int. Conference on Service-Oriented Computing (ICSOC), ACM Press (2004) 1–10
4. Spanoudakis, G., Zisman, A.: UML-based service discovery tool. In: Proceedings of the Int. Conference on Automated Software Engineering (ASE), IEEE Computer Society (2006) 361–362
5. Gardner, T.: UML modelling of automated business processes with a mapping to BPEL4WS. In: Europian Workshop on OO and Web Services (ECOOP). (2004)
6. Deubler, M., Meisinger, M., Rittmann, S., Krger, I.: Modelling crosscutting services with UML sequence diagrams. In: ACM/IEEE Int. Conference on Model Driven Engineering Languages and Systems (MoDELS). (2005)
7. Kramler, G., Kapsammer, E., Retschitzegger, W., Kappel, G.: Towards using UML 2 for modelling web service collaboration protocols. In: Proceedings of the Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA), Springer London (2005) 227–238
8. Feuerlicht, G., Meesathit, S.: Design method for interoperable web services. In: Proceedings of the Int. Conference on Service Oriented Computing (ICSOC), ACM Press (2004) 299–307
9. Lau, D., Mylopoulos, J.: Designing web services with tropos. In: Proceedings of the Int. Conference on Web Services, IEEE Computer Society (2004)
10. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Journal of Autonomous Agents and Multi-Agent Systems **8**(3) (2004) 203–236
11. Kazhamiakin, R., Pistore, M., Roveri, M.: A framework for integrating business processes and business requirements. In: Proceeding of the Enterprise Distributed Object Computing Conference, IEEE Computer Society (2004)
12. Penserini, L., Perini, A., Susi, A., Mylopoulos, J.: From stakeholder needs to service requirements. In: Proceeding of Int. Workshop on Service-Oriented Computing: Consequences for Engineering Requirements, IEEE Computer Society (2006)
13. Aiello, M., Giorgini, P.: Applying the tropos methodology for analysing web services requirements and reasoning about qualities of services. CEPIS Upgrade - The European journal of the informatics professional **5**(4) (2004) 20–26
14. Zimmermann, O., Schlimm, N., Waller, G., Pestel, M.: Analysis and design techniques for service-oriented development and integration. INFORMATIK (2005)
15. Arsanjani, A.: Service-oriented modelling and architecture (SOMA). Technical report, IBM developer, http://www.ibm.com/developerworks/webservices/library/ws-soa-design1 (2004)
16. Eriksson H-E., P.M.: Business Modelling with UML: Business Patterns at Work. Addison Wesley (1999)
17. Marcos, E., de Castro, V., Vela, B.: Representing web services with UML: A case study. In: Proceedings of the Int. Conference on Service-Oriented Computing (ICSOC), Springer (2003) 17–27