PROCEEDINGS OF THE 13th ANNUAL CONFERENCE ON WORLD WIDE WEB APPLICATIONS

14-16 September 2011
Johannesburg
South Africa

Editors:

A. Koch
P.A. van Brakel

Publisher:

Cape Peninsula University of Technology
PO Box 652
Cape Town
8000

Proceedings published at
http://www.zaw3.co.za

ISBN: 978-0-620-51918-2

TO WHOM IT MAY CONCERN

The full papers were refereed by a double-blind reviewing process according to South Africa's Department of Higher Education and Training (DHET) refereeing standards. Before accepting a paper, authors were to include the corrections as stated by the peer-reviewers. Of the 59 full papers received, 41 were accepted for the Proceedings (acceptance rate: 69.5%).

Papers were reviewed according to the following criteria:

- Relevancy of the paper to Web-based applications
- Explanation of the research problem & investigative questions
- Quality of the literature analysis
- Appropriateness of the research method(s)
- Adequacy of the evidence (findings) presented in the paper
- Technical (e.g. language editing; reference style).

The following reviewers took part in the process of evaluating the full papers of the 13th Annual Conference on World Wide Web Applications:

Prof S Berman
Department of Computer Science
University of Cape Town
Cape Town

Prof RA Botha
Department of Business Informatics
Nelson Mandela Metropolitan University
Port Elizabeth

Mr AA Buitendag
Department of Business Informatics
Tshwane University of Technology
Pretoria

Prof AJ Bytheway
Faculty of Business
Cape Peninsula University of Technology
Cape Town

Dr A Chigona
E-Learning Support and Innovation Unit
University of the Witwatersrand
Johannesburg

Prof T du Plessis
Department of Information and Knowledge Management
University of Johannesburg
Johannesburg

ISBN: 978-0-620-51918-2

ISBN: 978-0-620-51918-2

Prof JS van der Walt
Department of Business Informatics
Tshwane University of Technology
Pretoria

Prof D van Greunen
School of ICT
Nelson Mandela Metropolitan University
Port Elizabeth

Dr SC Warden
Faculty of Informatics and Design
Cape Peninsula University of Technology
Cape Town

Further enquiries:

Prof PA van Brakel
Conference Chair: Annual Conference on WWW Applications
Cape Town
+27 21 469 1015 (landline)
+27 82 966 0789 (mobile)

ISBN: 978-0-620-51918-2

The supportive role of service contract design for service governance

J. Chetty
Department of Applied Information Systems
University of Johannesburg
Johannesburg, South Africa
jacquic@uj.ac.za

M. Coetzee
Academy of Computer Science and Software Engineering
University of Johannesburg
Johannesburg, South Africa
marijkec@uj.ac.za

Abstract

Enterprise architecture is concerned with the design of an overall architectural vision for organisations. Service oriented architecture is a design paradigm that supports this vision, as applications are built from services that are composed by business processes. A service is a key element of service oriented architecture and represents technical solutions to business problems. The rules of engagement between services are expressed by service contracts in order to govern the execution of business processes. However, organisations develop services and their respective service contracts quickly, often without much thought to their management and maintenance. Services designed in this manner, do not promote governance. To gain an insight into service contract design, this paper describes the contract-first and the contract-last design approaches. The deficiencies of the contract-last design approach are highlighted. Service contract design requirements are identified, and governance is considered as a design requirement to consider. The paper finally proposes that the contract-first approach to service design should ideally be used to support service governance.

Keywords: Enterprise architecture, service oriented architecture, governance, service contract

1. Introduction

Service Oriented Architecture (SOA) (Erl, 2008) is a paradigm for organizing and utilising distributed capabilities that may be under the control of different ownership domains, and implemented using a variety of technology stacks. SOA is a holistic approach to designing systems in a distributed environment, where integration is mandatory. Organisations gain a competitive edge by exposing system capabilities or business functions as services, to be re-used for different applications and purposes. These services are well-defined, self-contained, and do not depend on the context or state of other services. Even though SOA is protocol independent, web

ISBN: 978-0-620-51918-2

services technology (Varia, 2011) is becoming the most common implementation of SOA.

While most organisations commence their SOA drive with a pilot project, they quickly begin initiatives that span multiple departments or business organisations. Services proliferate and if they are left ungoverned, an organisation may not be able to manage such services. Governance is needed to direct, control, manage and monitor the increasing number of services, in order to ensure reuse and consistency, and avoid duplication of work. A service contract can provide a supportive role for service governance between services, by describing aspects, such as security (Erl, 2008).

A service contract is an expression of the overall purpose and capabilities of a service in order to provide consumers with sufficient information about what the service does (Erl, 2006). It is the first document that a consumer has contact with before service interaction occurs. The service contract is shared amongst many consumers, all who have different needs and demands from a service. It is important that consumers understand and agree to the requirements of the service contract. The design of the service contract must therefore be given careful consideration.

There are different service contract design approaches that can be used. By designing service contracts using a particular approach, the governance of interactions between services and consumers can be supported. This paper analyses and evaluates the role of service contract design to determine whether such service contracts can assist with establishing governance. Two approaches, namely the contract-first and the contract-last design approach, are compared in this regard. Section 2 provides a background on SOA governance. Section 3 defines the service contract. Section 4 describes the most commonly used design approach, namely the contract-last design approach and highlights its deficiencies. To determine the requirements for service contract design, section 5 describes various design requirements and considers governance as a key element when designing a service contract. Section 6 describes the contract-first design approach and illustrates that many design requirements form part of this design approach. Finally, section 7 provides a conclusion.

2. SOA governance

Governance is a set of processes, policies, behaviours, laws and institutions (Skonnard, 2006). IT governance is considered a subset of governance (Christensen et al., 2007). It is a framework that consists of processes, organisational structures and leadership to ensure that IT systems align themselves with the strategies and objectives of the organisation (Skonnard, 2006). In its turn, SOA governance can be seen as an extension of IT governance (Christensen et al., 2007). SOA governance ensures that stakeholders define, implement and execute a business model and accountability framework for SOA. SOA governance also defines policies for aspects, such as security and quality of service (QoS). It introduces mechanisms to control the enforcement of such policies, and monitor

ISBN: 978-0-620-51918-2

whether service transactions behave in a predictable manner (Niemann et al., 2008). SOA governance is also concerned with *service governance*.

Whereas SOA governance concerns all SOA activities, service governance is a subdivision of SOA governance. It provides direction for the design, development, implementation, testing, quality assurance, management, maintenance, deployment, service registration, publishing and retirement of services (Marks, 2008). It ensures that services can be reused across domains of control. While SOA governance is responsible for ensuring that policies related to information security are stipulated, service governance conceptualizes, constructs, publishes and monitors such policies. Service governance is also seen as lifecycle governance, where it is divided into design-time governance and run-time governance (Marks, 2008). Activities, such as design and development form part of design time governance, while activities, such as publishing and maintenance form part of run time governance.

SOA governance is about defining policies that can control SOA processes. Service contracts can be used to define such policies.

3. Service contracts

Service contracts form the foundation for communication between services and therefore represent the most fundamental architectural element of an SOA (Erl, 2008). They support the relationship between services by establishing an agreement amongst services that are interacting with one another. While it is not required for the agreement to be entered into legally or to be explicitly negotiated (OASIS, 2008), the service contract must be precise and unambiguous. A service contract describes both the functional and non-functional requirements of a service.

Functional requirements describe what a service does (Erl, 2008) using schemas, such as XSD (Desmet et al., 2007) and WSDL (Clement et al., 2007). For schemas, the vocabulary describes the data to be exchanged between services (Erl, 2008) and the operations that are needed when messages are exchanged. WSDL is a thin layer over XSD that defines service endpoints, service operations, input and output messages supported by each operation and the data representation model of each message's content. These requirements dictate the rules under which interaction between services may take place.

Non-functional requirements include the rules and constraints that govern the interaction of service operations, such as security, service level agreements and reliability, expressed as policies. Business-level characteristics, such as legal requirements, not fundamental to the service interaction, can also be included. Various policies can be applied to a service at the same time and are attached to its service contract. Policies are combined to form an applicable policy, which parties must agree to before service interaction takes place.

Functional and non-functional requirements provide an organisation with an opportunity to govern service interaction. For example, a service contract may include a business rule, which states that a username / password combination must

ISBN: 978-0-620-51918-2

be used when accessing a specific service. This rule allows an organisation to ensure that a service is secured according to business requirements. The service contract can promote governance as all service interaction can automatically be controlled, managed and monitored (Von Solms & Eloff, 2009) by governance components according to these rules.

Service contracts can be designed according to either a contract-last or contract-first design approach that may affect the quality of the service contract. This in turn can determine whether service contracts are able to provide a governance role. The contract-last design approach is discussed next.

4. Contract-last design approach

In the contract-last design approach, implementation classes or service logic are developed first. From these classes, the functional requirements, such as data types, operations and messages are generated using design tools, such as Microsoft Visual Studio (Erl, 2008). Even though it is a design approach that is not recommended, it is widely used (Erl, 2008). The reason for its popularity is that service contracts can be developed quickly, with tools generating the bulk of the service contract. This approach is therefore popular with inexperienced developers. The problems created by this approach are as follows:

- As the developer has no control over the content of the service contract it may contain ambiguities that are difficult to interpret by consumers, leading to non-compliance.
- As the content is auto-generated, changes to the service contract are difficult to apply.
- When developers alter the service contract to reflect rules correctly, a version change occurs that may lead to consumers using the incorrect service contract (OASIS, 2008).

The main deficiency of the contract-last design approach is the lack of control that the developer has over the design of the service contract. These service contracts may not include elements, such as business rules, specific organisational policies, security requirements, and best practices. The complexities associated with not being able to control, manage and monitor such service contracts needs to be considered. In order to be able to design a service contract that supports service governance, certain requirements need to be considered.

5. Requirements for service contract design

Developing a service contract that supports service governance increases the complexity of designing service contracts. This research now proposes that the following set of requirements need to be considered:

- base the service contract on business requirements;
- use of best practices;
- use of industry standards;
- apply all functional requirements;

ISBN: 978-0-620-51918-2

- express non-functional requirements; and
- manage and control versioning.

Each service contract design requirement is now discussed.

a) Base the service contract on business requirements

A service contract must be built on business requirements as they reflect the objectives and processes of the organisation (Weil & Ross, 2004). The goal of SOA governance is to ensure that the IT systems of organisations align themselves with the strategies and objectives of the organisation (Erl, 2006). For an organisation to fulfil this governance obligation, service contracts need to be based on business requirements.

b) Use of best practices

A best practice is a set of specifications that has been developed by experts with knowledge in a particular area (Von Solms & Von Solms, 2006). Cobit (IT Governance Institute, 2007) is an example of a best practice that can be used as a guideline (Von Solms & Von Solms, 2006) to develop service contracts. Cobit is an IT governance framework that can, for instance, help an organisation to determine the level of security and control of IT systems (IT Governance Institute, 2007). For example, Cobit states that adequate identity management must be in place to prevent unauthorised access of computer resources. A service contract that complies with this requirement could provide support for service governance. It is imperative that developers are made aware of best practices in order to allow them to develop service contracts that comply to these practices.

c) Use of industry standards

The service contract that is developed for web services should be based on standards, such as SOAP (Alves et al., 2007) and WS-Security (Desmet et al., 2008). A framework, such as WS-Security allows services to express rules and constraints, such as identity management (Erl, 2008) in a standard manner that is endorsed by standards organisations, such as OASIS (IT Governance Institute, 2007). By designing a service contract using standards, it is ensured that the service contract follows all required schema and policy rules, as industry standards provide a formal way in which rules can be represented. Applying these standards also means that services, which are designed in a standard format, can easily interact with one another. This promotes better interoperability and agility amongst services.

d) Apply all functional requirements

The purpose of a service contract is that it must be well defined and extensible. As explained previously, tools should not be used to auto-generate these documents as this reduces the amount of control (Skonnard, 2006) that a developer has over the service contract. A service contract that explicitly states all functional requirements, such as service endpoints, service operations, input and output messages, is a service contract that is developed using a step-by-step approach. Quality assurance

ISBN: 978-0-620-51918-2

checks and testing can be completed at regular intervals (Tost, 2010). Service governance is a step-by-step process, which controls the lifecycle of a service from conception to deployment (Marks, 2008). Quality assurance checks can be completed as service development progresses. However, services that are auto-generated may not be well developed, as these tools may view certain functional requirements as optional. Quality assurance and testing of the service may not be adequately completed (Erl, 2008).

e) Express non-functional requirement

SOA governance is about defining which decisions need to be made and service governance is about ensuring that those decisions can be implemented, maintained and managed (Tost, 2010). For an SOA, these decisions and business rules are expressed and enforced through a well-documented set of rules and guidelines, known as policies (Kanneganti & Chodavarapu, 2008). Capturing such non-functional requirements as policies, allows an organisation to separately implement, maintain, and manage such policies in a controlled environment.
Policies are central to SOA governance (Marks, 2008) (Tost, 2010) and they provide a hierarchy (Tost, 2010), where business requirements develop into business policies. These business policies are then translated into operational policies, which are machine-readable expressions. For example, a business requirement may state that an organisation must comply with all laws and regulations. The business policy translates this into a number of policies, such as “keep all consumer data private”. The operational policy then translates this into “ensure access control via username/passwords for sign-on” and “encrypt all consumer data”. These English statements are translated into machine-readable expressions. WS-Policy is a standard that is commonly used. For an SOA, policies provide an opportunity to establish standards and control mechanisms so that organisations can carry out their roles and responsibilities.

f) Manage and control versioning

For service interaction to occur, version control of service contracts need to be governed carefully. For example, a consumer may receive a mandatory datatype that consists of a surname and an initial. If this datatype changes to surname, middle name and an initial, the rules of engagement have been altered. The consumer may not be able to interact, if these rules change. There are various aspects to consider when service contracts need to be updated, namely:

- Policy changes should only be made by authorized individuals, as a change to a policy may impact service interaction, causing services to be disrupted (Erl, 2008), which in turn can generate incorrect data.
- Older versions of the service must remain valid so that consumers are not forced to change versions (Erl, 2008). These older versions must also be kept for auditing purposes.
- Service contracts that are developed using a tool can complicate versioning (Skonnard, 2006). For example, a consumer may be forced to use a service contract that is not compatible with its needs. This arose because a developer was unable to make changes to the service contract as it was automatically generated.

ISBN: 978-0-620-51918-2

- The importance of adequately establishing service governance to control the creation, implementation and enforcement of service contracts (Erl, 2008). Adequate service governance can be established by introducing service contract requirements, which specify who controls changes made to service contracts.

This section has described some design requirements that need to be taken into consideration when designing a service contract to support service governance. The following section identifies whether the contract-first design approach considers these requirements.

6. The contract-first design approach and service governance

The contract-last design approach produces immediate benefits, such as service contracts that are developed quickly. The long-term effect is that these service contracts do not encourage interoperability; are not expressive; and may not be in alignment with business requirements and standards (Dan et al., 2008). This section discusses the contract-first design approach. It illustrates that this design approach meets many of the service contract design requirements and service governance considerations needed for good service contract design.

A contract-first design approach applies service oriented design principles (Erl, 2008) to develop a service contract, which is based on business requirements (Varia, 2011). The service contract is developed first by designing a generic service interface to cater for future changes, and schemas and vocabularies to define all messages and data. It is then approved and imported into the development environment, where the service logic is created to support the service contract (Leelarnthne, 2009).

The contract-first design approach is not a very popular approach as it is tedious to begin with (Skonnard, 2006). From the developer's perspective, it does seem easier to develop the service logic and allow the service contract to be generated from the service logic. If the organisation is more concerned with productivity as opposed to interoperability, then the contract-last design approach is the design approach that can be used (Skonnard, 2006). Although the contract-first design approach seemingly takes more effort to develop service contracts, it consists of many benefits. These benefits are:

- Service contracts that reflect business requirements;
- Descriptive, standardized service contracts;
- Service contracts and compliance;
- Collaboration; and
- Version control.

The following paragraphs describe these benefits and include the manner to which each benefit promotes service governance.

a) Service contracts that reflect business requirements

ISBN: 978-0-620-51918-2

The contract-first design approach is based on developing the service contract from business requirements. A service contract that is developed from business requirements may consist of a more accurate representation of what the business requires (Varia, 2011). Service contracts that reflect business requirements promote service governance, as:

- they are aligned with the objectives of IT governance and ensure that business processes collaborate with IT systems;
- service logic is developed from the service contract (Varia, 2011), so the service logic reflects business requirements; and
- changes to the service contract are always made first and then changes to the service logic is made.

b) Descriptive, standardized service contracts

Standardized service contracts are contracts that consist of functional and non-functional components. Standardized service contracts that are developed using the Web services platform is becoming commonplace (Erl, 2008). Service contracts that are standardized promote service governance, as:

- they establish policies, which are well-documented rules and guidelines that implement decisions made by organisations;
- they establish standards, which promote interoperability among consumers;
- they are easier to direct, control, manage, monitor and audit;
- they provide an opportunity to form part of a standardized governance framework; and
- they provide an opportunity to ensure that information security is implemented.

c) Service Contracts and compliance

Compliance means that formal monitoring and auditing tools are in place to provide evidence that laws, regulations, and contractual arrangements amongst business processes are being adhered to (IT Governance Institute, 2007). Service contracts promote service governance, as:

- non-functional requirements that form part of a service contract can stipulate rules pertaining to contractual arrangements between business processes.

d) Collaboration

Collaboration means that developers from different domains communicate with each other when developing service contracts. The service contracts then provide a foundation on which service interaction amongst services, from different domains, can take place. Service contracts that encourage collaboration promote service governance, as:

- service contracts developed in collaboration with one another encourage standardization; and
- both parties agree to mandatory service contract rules and this means that it is easier to control and manage service contracts between multiple parties.

ISBN: 978-0-620-51918-2

e) Version control

Version control is about tracking and controlling changes made to software (Fogel, 2010), such as service contracts. A service contract that is designed, developed and implemented for the first time is allocated a version number, such as “inventory1.0”. Any change to the service contract requires a version update. Organisations that perform version updates in a methodical, controlled and managed manner perform version control. Managing service contract versioning means that consumers are only exposed to service contracts that apply to them. Service contracts that perform version control promote service governance, as these contracts can be controlled, managed and monitored to:

- ensure that business processes are connected to relevant service contracts; and
- ensure that service contract changes do not adversely affect business processes.

In summary, there are benefits to using the contract-first design approach. These benefits often depend upon the vision, philosophy and culture of an organisation. Table 1 compares the contract-first and contract-last design approach, so that organisations can decide which design approach is applicable to their vision and culture.

Although there is no right or wrong way in which to design service contracts (Skonnard, 2006), it is important that careful consideration is given to service contract design. As reflected in the paper, the manner in which service contracts are designed can positively or negatively influence the extent to which service governance is established. The contract-first design approach applies governance principles to the design of services. For example, developing services from business requirements, making use of standards, including policies, complying with rules and constraints and ensuring that the information security of the organisation’s assets are secure, is fundamental to developing governance. The contract-first design approach includes many of these governance requirements. It is a design approach that can be used to provide and support service governance.

Table 1: Comparison of the contract-first and contract-last design approach

Service design requirement	**Contract-last design approach**	**Contract-first design approach**
Base the service contract on business requirements	N	Y
Use of best practices	N	N
Use of industry standards	N	Y
Apply all functional requirements	N	Y
Express non-functional requirement	N	Y
Manage and control versioning	N	Y
Build the service logic based on service contract	N	Y

ISBN: 978-0-620-51918-2

7. Conclusion

Service contracts are created for specific consumers or groups of consumers and these service contracts are agreed upon. Service contracts may provide an opportunity to govern the relationship between services. For instance, rules that affect security can form part of the service contract. Once these rules are agreed to, consumers and services need to abide by these rules. This can provide compliance to the service contract. The design of a service contract is important as a lack of design can lead to a decrease in service governance.

This paper has discussed two service contract design approaches. Service contract design is a challenge as organisations have to rely on varying specifications to create a service contract. Although an organisation needs to decide which approach to make use of, the contract-first design approach does provide a supportive role for service governance. It is the right approach to use if an organisation considers reuse, interoperability and governance as an integral part of their SOA environment.

8. Acknowledgment

The financial assistance of the National Research Foundation (NRF) in South Africa under Grant number 66302 is hereby acknowledged. Any opinion, findings and conclusions or recommendations expressed in this material are those of the authors and the NRF does not accept any liability thereto.

9. References

Alves, A., Arkin, A., Askary, S., Barreto, C., Bloch, B., Curbera, F., Ford, M., Goland, Y., Guízar, A., Kartha, N., Liu, C.K., Khalaf, R., König, D., Marin, M., Mehta, V., Thatte, S., van der Rijn, D., Yendluri, P. & Yiu, A. (Editors). (2007). Web Services Business Process Execution Language Version 2.0. Available from: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html. (Accessed 10 March 2008).

Christensen, E., Curbera, F., Meredith, G. & Weerawarana, S. (2001). Web Services Description Language (WSDL) Version 1.1. Available from: http://www.w3.org/TR/wsdl. (Accessed 14 November 2007).

Clement, L., Hately, A., Rogers, T. & von Riegen, C. (Editors). (2004). OASIS: UDDI Version 3.0.2. Available from: http://uddi.org/pubs/uddi_v3.htm (Accessed 17 April 2008).

Leelarathne, D. (2009) Contract-First Design. Available from: http://wso2.org/library/articles/soa-contract-first-design (Accessed 8 May 2008)

Dan, A., Franck, R., Keller, A., King, R.P. & Ludwig, H. (Editors). (2003). Web Service Level Agreement (WSLA) Language Specification. Available from:

http://www.research.ibm.com/wsla/WSLASpecV1-20030128.pdf. (Accessed 20 February 2008).

Desmet, L., Joosen, W., Massacci, F., Philippaerts, P., Piessens F., Siahaan, I. & Vanoverberghe, D. (2008). Security-by-contract on the .net platform, Science Direct, pp. 25-32, Feb., 2008

Erl, T. (2006), Service Oriented Architecture: Concepts, Technology, and Design. New York: Prentice Hall

Erl, T. (2008), SOA Principles of Service Design: Indiana:Prentice Hall

Fogel, K. (2010) How to run a successful Free Software Project. Available from: http://producingoss.com/en/vc.html (Accessed August 2011)

Hallam-Baker, P., Kaler, C., Monzillo, R. & Nadalin, A. (Editors). (2006). Web Services Security: SOAP Message Security 1.1. Available from: http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf. (Accessed 10 January 2008).
Hawkins, J.M., (compiled by), (2001). The South African Oxford School Dictionary. 5th edition. Cape Town: Oxford University Press.

IT Governance Institute. (2007). Cobit 4.1. Illoinois: IT Governance Institute.

Jencmen, A. & Yehudai, A. (2006), Fortified Web Services Contracts for Trusted Components, Computer, pp. 919-926, Sep., 2006

Kanneganti, R. & Chodavarapu, P. (2008). SOA Security. Manning

Marks, E.A., (2008). Service-Oriented Architecture Governance for the Services Driven Enterprise. Wiley

Marks, E.A. & Bell, M. (2006), Service-oriented Architecture A Planning and Implementation Guide for Business and Technology. New Jersey:Wiley

Niemann, M., Eckart, J. & Steinmetz, R. (2008), Towards a Generic Governance Model for Service-oriented Architecture. AMCIS, 2008

OASIS SOA Reference Model Technical Committee (2006). Policies and Contracts. Available from: http://wiki.oasis-open.org/soa-rm/TheArchitecture/PoliciesAndContracts (Accessed 25 January 2008).

Skonnard, A., (2006). Contract First Service Development. Available from: http://msdn.microsoft.com/en-us/magazine/cc163800.aspx. (Accessed 01 June 2008)
Von Solms, S.H. & Von Solms, (2006). Information Security

Tost, A., (2010). SOA Governance Using Policies. Available from www.soasymposium.com (Accessed July 2011)

ISBN: 978-0-620-51918-2

Varia, J. (2011), Architecting for the Cloud: Web services Best Practices. Available from: http://jineshvaria.s3.amazonaws.com/public/cloudbestpractices-jvaria.pdf (Accessed Aug 2011)

Windley, P.J., (2006). SOA Governance: Rules of the game. Available from: http://infoworld.com. (Accessed 10 May 2008)

Von Solms, S.H. & Eloff J.H.P., (2009). Information Security Governance

Weil, P & Ross, R., (2004). IT Governance. Harvard Business School Press

ISBN: 978-0-620-51918-2