Information and Software Technology xxx (2011) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

SOAdapt: A process reference model for developing adaptable service-based applications

Stephen Lane [a,*], Antonio Bucchiarone [b], Ita Richardson [a]

[a] Lero, The Irish Software Engineering Research Centre, University of Limerick, Ireland
[b] Fondazione Bruno Kessler, Via Santa Croce 77, 38100 Trento, Italy

ARTICLE INFO

Article history:
Received 15 April 2011
Received in revised form 24 October 2011
Accepted 25 October 2011
Available online xxxx

Keywords:
SOA
Service-based application
Software process
Process reference model

ABSTRACT

Context: The loose coupling of services and Service-Based Applications (SBAs) have made them the ideal platform for context-based run-time adaptation. There has been a lot of research into implementation techniques for adapting SBAs, without much effort focused on the software process required to guide the adaptation.
Objective: This paper aims to bridge that gap by providing an empirically grounded software process model that can be used by software practitioners who want to build adaptable SBAs. The process model will focus only on the adaptation specific issues.
Method: The process model presented in this paper is based on data collected through interviews with 10 practitioners occupying various roles within eight different companies. The data was analyzed using qualitative data analysis techniques. We used the output to develop a set of activities, tasks, stakeholders and artifacts that were used to construct the process model.
Results: The outcome of the data analysis process was a process model identifying nine sets of adaptation process attributes. These can be used in conjunction with an organisation's existing development life-cycle or another reference life-cycle.
Conclusion: The process model developed in this paper provides a solid reference for practitioners who are planning to develop adaptable SBAs. It has advantages over similar approaches in that it focuses on software process rather than the specific adaptation mechanism implementation techniques.

1. Introduction

The Service-Oriented Computing (SOC) paradigm is centered around the publication and consumption of loosely coupled computational elements known as services. These software services are often owned and controlled by third parties rather than the application developers who consume them. A key benefit of SOC is the ability to rapidly compose distributed software services into useful Service-Based Applications (SBAs). These applications are able to offer complex and flexible functionalities in widely distributed environments by composing different types of services. Such services are often not under the control of systems developers, but they are simply exploited to obtain a specific functionality.

The possibility of rapidly composing SBAs opens up new opportunities for conducting ad hoc business transactions. However, with this new freedom there are also new challenges, for example, when services are owned and controlled by third parties their reliability or availability are not guaranteed. Adaptable SBAs change their behavior, reconfigure their structure and evolve over time reacting to changes in the operating conditions, so as to continuously meet users' expectations. This concept is fundamental since those systems live in distributed and mobile devices which have frequently changing environments. Also, user goals and needs may change dynamically, and systems should adapt their functionalities accordingly without intervention from technicians.

When designing and developing traditional software, modeling and implementing the application's logic is usually the only problem to consider. With SBAs, adaptability should also be considered during the requirements engineering, design and construction phases of SBA development. Similarly the adaptation of an SBA needs to be monitored and controlled when these applications are in operation.

Many of the existing SBA development methodologies are based on the results carried out in the fields of classical software and system engineering and do not easily facilitate SBA adaptation [1–3]. Some of the reported SBA development approaches such as SOUP (Service Oriented Unified Process) [4], ASTRO [5] or an approach by Linner et al. [6] do support some level of adaptation, however, lack sufficient process details. Lane and Richardson [7] carried out a systematic literature review of SBA development approaches,

* Corresponding author.
E-mail addresses: stephen.lane@lero.ie (S. Lane), bucchiarone@fbk.eu (A. Bucchiarone), ita.richardson@lero.ie (I. Richardson).

0950-5849/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.infsof.2011.10.003

they identified 57 such approaches of which there were only eight that specifically dealt with adaptation. Only four of the eight approaches were concerned with adaptation of SBAs, the others were concerned with the adaptation of services.

Each of the four approaches show interesting features, but even those that enable the definition of various adaptation strategies lack a coherent design approach to support designers in this complex task. Moreover, they tend to focus on the implementation process without considering what impact adaptation has on the rest of the development and operational life-cycle [8–11]. Finally, they also tend to focus on particular types of adaptation, such as adaptation due to requirements violations [12], or substitution services due to application constraints [13], so it is difficult to elicit generic adaptation specific processes from them. In summary, each of these four approaches focused on the analysis and design processes without consideration for any other development or runtime processes. The solution proposed in [14] can be considered as a first step in this direction. The authors of this paper have proposed a process model for SBAs where adaptation is a first class concern. This process model provides a number of steps that, if implemented during the life-cycle of a SBA, will allow the SBA to adapt in response to changing runtime contexts.

1.1. Objectives

The primary objective of this paper is to develop an empirically grounded process model that can be used as a reference for service practitioners who want to develop adaptable SBAs. The process model will focus on activities related to adaptation only, meaning that it will have to be used in conjunction with an other SBA development life-cycle. In order for an SBA to be adaptable, certain adaptation mechanisms need to be put in place when the application is initially developed. These mechanisms then facilitate adaptation when the application is in operation.

There are two types of adaptation, static adaptation and dynamic adaptation [15]. With static adaptation, adaptation mechanisms are hard coded into the application at development time and the adaptation logic cannot be changed without recoding. Dynamic adaptation on the other hand allows the adaptation logic to be modified or replaced during runtime without shutting the system down. Dynamic adaptation is more flexible than static adaptation but it requires some process to guide the manual intervention during runtime.

Process details will be elicited from relevant SBA development and adaptation approaches as well as qualitative data gathered from interviews with industry practitioners. The validity of the SBA adaptation process model will be demonstrated through a comparison with a previously validated adaptation process model. This comparison will show that the process model contains the minimum set of activities required for runtime software adaptation. The minimum set of activities for adaptation exclude activities that are not specifically related to adaptation. The processes and activities from the process model will also be mapped to an empirically based SBA development life-cycle. This will illustrate whether or not the model is suitable for use in a real world SBA development context.

1.2. Taxonomy

The terminology used in the areas of software process and service engineering research are often ambiguous and conflicting. In order to define what is meant by the more common software process and service engineering terms used in this paper a brief outline will be given for each. The definitions for the service engineering terms are based on the S-Cube knowledge model [16] while the software process terms are based on Derniame et al. [17].

Service-based applications are composite applications composed of multiple software services. Software services or services are computational elements that expose their functionality over computer networks. Because services are self-contained and loosely coupled it is possible to build SBAs from unrelated services that may or may not be in the control of the SBA developers. An example SBA is the travel booking application, which is composed of three third party services: a car rental, flight reservation and hotel reservation services. These three unrelated services are composed to provide useful SBA, synonyms for SBAs include service-oriented systems or service-based systems.
Adaptable service-based applications. ASBAs are a specific type of SBA that adapt during operation by following built-in adaptation strategies. Adaptation of SBAs is facilitated by monitoring mechanisms that monitor changes in monitored properties. Adaptation may be desirable for many reasons such as changes in quality characteristics, change in business requirements or change in the cost of services being consumed. Adaptation may occur through the selection of alternative services from pre-defined lists or through discovery of new services from service directories.
Software process. Software process refers to the set of processes and activities undertaken to develop software. This process is often measured so that it can be managed or improved. Some software developers follow predefined processes while others have a more ad hoc software process. Nevertheless, whether defined or not, each software development project will follow a particular set of processes. Synonyms for software process include: software development process or software life-cycle.
Process models are a documented representation of a set of real world processes, activities and their interrelations. A *software process model* is specific type of process model for software development. Creating a process model for a software development process allows reasoning about the model so that it can be improved. Ideally a process model should perfectly articulate the real world processes and activities that they are attempting to describe.
Process reference models are best practice or exemplar *process models* that can be used as a reference by individuals or groups to improve on their own processes. A specific type of process reference model is a *software process reference model* which is an exemplar process model used for developing software, for example, the Capability Maturity Model Integration (CMMI) and ISO-15504 document process reference models.
Life-cycle models in software engineering are *process models* that define the entire software development life-cycle. This is different from a *process model* which may only address particular parts of the development cycle.
Life-cycle process is a collection of software development activities that occur in sequence and make up one logical part of the software development life-cycle. For instance, "Requirements engineering" is a *software life-cycle process* that may contain activities such as "Elicit requirements" or "Develop requirements specifications".

2. Background

2.1. A life-cycle to develop adaptable SBA

The life-cycle model shown in Fig. 2, proposed by the S-Cube consortium [18], highlights the typical design-time iteration cycle that leads to the design of adaptable SBAs. It introduces an iteration cycle at run-time that is undertaken in all cases in which

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

the adaptation needs are addressed on-the-fly. The two cycles coexist and support each other during the lifetime of the application. In particular, the design-time activities allow for *evolution* of the application, that is, for the introduction of permanent and, usually, important changes, while the run-time activities allow for temporary *adaptation* of the application to the specific circumstances that are occurring at a certain time.

2.2. Design guidelines for adaptation

The design of SBAs with runtime adaptation capabilities may be motivated by variety of needs. Such needs may concern the component services or the context of SBAs. These include:

Changes in the service functionality due to variation of the service interface (e.g., signatures, data types, semantics), variation of service interaction protocol (e.g., ordering of messages) or Application failures.
Changes in the service quality due to service availability, degrade of QoS parameters, violation of SLA or decrease of service reputation (e.g., black lists), etc.
Changes in the service context as a result of changes in the business context, changes in agile service networks, or new business regulations and policies.
Changes in the computational context such as different devices, protocols, and networks.
Changes in the user context such as different user groups and profiles, social environment, physical settings (e.g., location/time), and different user activities.

Some of these aspects may be interleaved. For example, if a user moves to a new location (change in the user context), a new set of services may be available (change in the business context) with different bandwidth (change in the computational context). Each factor can be associated with a set of adaptation strategies that are suitable to re-align the application within the system and/or context requirements. In order to select the adaptation strategy which should be applied, it is necessary to consider that adaptation needs may be associated with other requirements that are important for designing and performing adaptation. In particular, the scope of the change (whether the change affects only a single running instance of the SBA or influences the whole model) and the impact of the change (the possibility of the application to accomplish its current task) should be considered. Depending on these parameters different strategies may apply.

Among the adaptation strategies, it is possible to distinguish domain-independent or domain-dependent strategies. The former are applicable in almost every application context while the adoption of the latter is limited to specific execution environments. Table 1 defines the most common domain-independent adaptation strategies.

The identification of the most suitable strategy is supported by a reasoner that also bases its decisions on multiple criteria extracted from the current situation and from the knowledge obtained from previous adaptations and executions. After this selection, the enactment of the adaptation strategy is automatically or manually performed. The execution of all activities and phases in all runtime phases may be performed autonomously by SBAs or may involve active participation of the various human actors.

2.3. Component-based software engineering

Component-Based Software Engineering (CBSE) is often described as the precursor to service-oriented computing [19]. In CBSE software applications are composed of self contained software components that contain all of the functionality necessary for a system of business process. In component-based applications communication occurs through interfaces that are described with some interface description language. Components do not have to be implemented using the same programming languages providing there is some intermediary mechanism that can serialize data and objects between components.

In several Service-Oriented Software Engineering (SOSE) methodologies components are first implemented before being exposed as services over a network [3]. This shows the high degree of overlap between the two development paradigms. Although, services and components may differ on their levels of granularity. For example, software components are usually composed of more fine grained objects while in turn they often form the basis of more coarse grained services [20].

When looking at service-oriented adaptation techniques it may first be beneficial to review related component-oriented approaches given the relationship between the two paradigms. Oreizy et al. [21] have proposed such an approach for component-based adaptation. Their approach contains two cycles that run in parallel, and evolution management cycle and an adaptation management cycle. The evolution management cycle manages modifications to the application through the use of tools and adaptation mechanisms. Modifications permitted include the addition, removal or replacement of application components and their connectors. The evolution management cycle examines changes to the application and prevents and changes that might make the operation of the application unsafe or inconsistent. The adaptation management cycle on the other collects observations while the application is running. These observations are then analyzed in order to determine if it is necessary to adapt the application. Should

Table 1
Description of the most common domain-dependent adaptation strategies.

Adaptation strategy	Description
Service substitution	Reconfiguration of the SBA with a dynamic substitution of the a service with another one
Re-execution	The possibility of going back in the process to a point defined as safe for redoing the same set of tasks or for performing an alternative path
(Re-)negotiation	Simple termination of the service used on the requester side and re-negotiation of the SLA properties to complex management on reconfiguration activities on the provider side
(Re-)composition	Reorganization and rearrangement of the control flow that links the different service components in the business application
Compensation	Definition of ad hoc activities that can undo the effects of a process that fails to complete
Trigger evolution	Insertion of workflow exception able to activate the application evolution
Log/update adaptation information	Storage of all the information about the adaptation activities for different goals (e.g., service reputation, QoS analysis, outcome of adaptation)
Fail	The system reacts to the changes by storing the system status and causing the failure of the service and re-executing it

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

an adaptation be deemed necessary, then the adaptation cycle provides the necessary tools for deploying alternative components.

3. Research method

There were two phases involved in the construction of the process model during this study. The first phase involved the development of a Frame of Reference (FoR), named SOAdapt-FoR, based on relevant literature. This was enhanced with empirical evidence gathered during the second phase of the study, resulting in the SOAdapt process model. The research design largely follows Ahlemann and Gastl's [22] process for empirically grounded reference model construction.

3.1. Construction of the SOAdapt-FoR model

The SOAdapt-FoR model was constructed by analysing existing publications that contain activities for adapting SBAs. Both peer-reviewed and non-peer-reviewed publications were analyzed to compile these process details. One of the key sources of adaptation activities was a Systematic Literature Review (SLR) of SBA development process models. The SLR, that was carried out by Lane and Richardson [7], identified which of the process models supported SBA adaptation. Many of the activities identified were similar and at varying levels of abstraction. In order to eliminate duplication and to show the activities at the same level of abstraction, each activity was assigned a generic name. Similar activities were assigned the same generic name, resulting in a smaller list of unique generic activities.

In their Systematic Literature Review (SLR), Lane and Richardson identified 8 publications which contain high-level processes and activities for adapting SBAs. Similarly Lane et al. [23] identified several additional publications, during an ad hoc literature search, which contained SBA adaptation activities. Unfortunately, most of the approaches identified the in SLR [7] and the ad hoc search [23] are non-holistic and lacking the required processes to fully guide SBA adaptation. Each of these publications as well as two other papers from Bucchiarone et al. [24,14], were searched, resulting in a total of 50 SBA adaptation activities (see Appendix A).

There is a wide variety of process modeling notations such as the Business Process Modeling Notation[1] (BPMN) or the Unified Modeling Language[2] (UML), and domain specific process meta-models such as the Software and Systems Process Engineering Meta-Model[3] (SPEM) of the meta-model described in ISO/IEC 15288 [25]. BPMN and UML both facilitate the graphical representation of process models which is not the goal of this exploratory research. SPEM is a very detailed meta-model which also uses UML in order to instantiate process models. Rather than these graphical modeling approaches, the aim of this research is to document key process attributes such as activities, tasks, artifacts and stakeholders. The process meta-model defined in ISO/IEC 15288 contains these key process attributes.

The "activity" abstraction level, as defined in ISO/IEC 15288, was chosen for the generic activity names. ISO/IEC 15288 defines activities as a group of related tasks that are required to achieve the outcomes of a process. Using UML class diagram notation, Fig. 1 [25] illustrates the ISO/IEC 15288 process constructs and their interrelations. The figure shows that a Process can have zero or more Sub-Processes and one or more Activities. In turn, it shows that an Activity can have one or more tasks which can have zero or more Notes. It also shows that each process has a Name, a Purpose, and an Outcome. The meta-model will be slightly adapted for this research with the removal of the process outcomes and the addition of stakeholders and artifacts. Stakeholders are the individuals that participate in the activities of a process while artifacts are the physical entities such as code or documentation that are produced by or modified by activities.

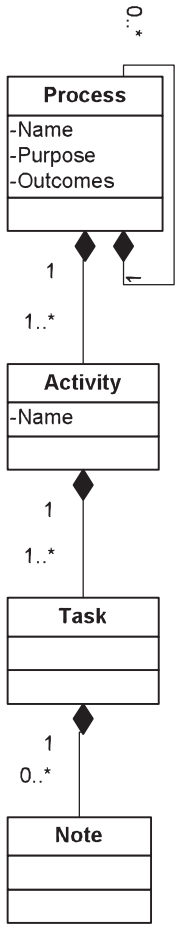

Fig. 1. ISO/IEC 15288 process constructs.

The S-Cube life-cycle, illustrated in Fig. 2 [18], was chosen as a starting point to develop the SOAdapt-FoR model. The S-Cube life-cycle is a high-level life-cycle model skeleton proposed for the development of adaptable SBAs. The generic adaptation activities identified were mapped to the appropriate processes of the S-Cube life-cycle resulting in the SOAdapt-FoR model. This approach was taken in order to leverage the existing research which has been conducted by the S-Cube consortium. Additionally, the S-Cube life-cycle covers each aspect of the development cycle without being too detailed. This allows it to be easily modified and enhanced with activities reported in the service engineering literature. The SOAdapt-FoR model is a high-level model and only uses the Process and Activity constructs as defined in ISO/IEC 15288.

3.2. Construction of SOAdapt

SOAdapt was constructed with data gathered from an expert-opinion survey and an interview-based case study. This data was then used to modify and extend SOAdapt-FoR. Therefore process attributes discovered during the interviews that were not present in SOAdapt-FoR represent the gaps in the literature. Altogether there were 198 passages of interview transcript which were assigned codes because they contained information about adaptation

[1] http://www.bpmn.org/.
[2] www.uml.org.
[3] http://www.omg.org/spec/SPEM/.

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

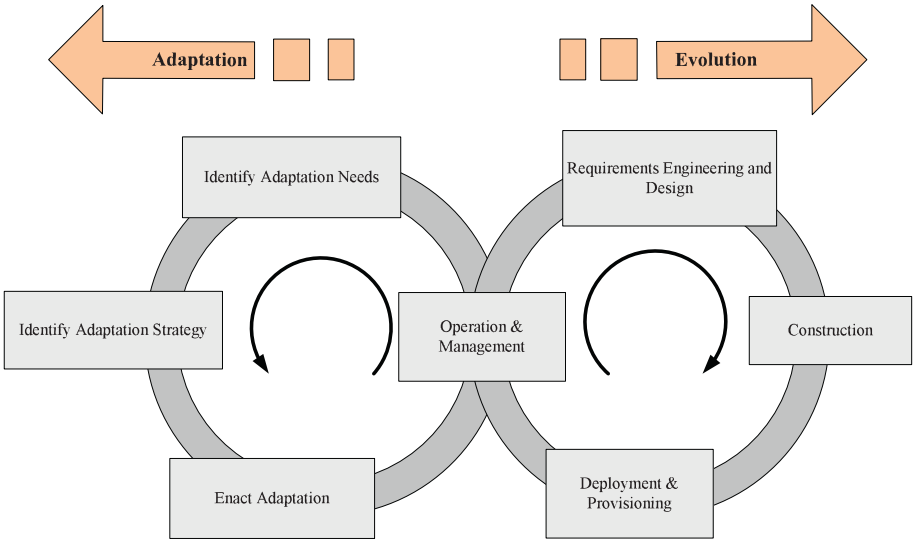

Fig. 2. The S-Cube life-cycle of adaptable SBAs.

processes, activities, tasks, stakeholders, or artifacts. Additionally there were 12 passages of text which were coded because they contained information about the background's of interview participants.

3.2.1. Expert opinion survey

The first stage in the expert-opinion survey was the development of an interview guide for use during the survey interviews. The interview guide was divided in two parts: the first part deals with the participants backgrounds and their roles, while the second part focuses on the elicitation of adaptation activities. Participants discuss the activities within the SOAdapt-FoR model, and to provide feedback. The respondents were asked whether or not they agreed with the activities contained in the SOAdapt-FoR model, and what changes they would make. Then they were asked to describe the tasks required to complete these activities, the stakeholders involved and the artifacts produced by the activities.

The interviews conducted were semi-structured interviews [26]. Interview guides for this type of interview contain open-ended questions which allow interviewers to ask follow on questions when necessary. This flexibility allows the interviewer to ask the interviewees detailed questions about their specific areas of expertise. This methods allowed the model to be expanded in depth as well as breath, existing processes received much more detail while new processes were also added. Background questions were asked to provide context information for each of the participants. Each of the participants were asked back ground questions such as:

- What is you current position?
- Do you work with adaptable Service-Based Applications (SBAs)?

Then they were asked to identify adaptation process attributes that could be used to develop the SOAdapt-FoR further. For example, for each of its activities the were asked:

1. At what stage in the SBA development life-cycle should it occur, or should it occur at all?
2. What stakeholders should be involved in the activity?
3. What lower level activities or tasks should it contain?
4. What artifacts are required by or produced by the activity?

They were asked to describe the SBA development processes within their organisations, focusing specifically on there individual roles, where they could provide the most insight. These questions were used to elicit details of processes and activities that were not contained in SOAdapt-FoR.

A pilot interview was conducted using this interview format. This results of the pilot showed that this approach struck a good balance between flexibility and structure. A strictly structured questioning approach would have prevented the interviewer from probing concepts that were not considered while developing the SOAdapt-FoR model.

Once the interview guide was completed and piloted, the next step was to identify and contact suitable interview partners. This proved to be one of the most challenging aspects of the study given the specialized nature of the process model being constructed. In order to participate, interview partners needed to be experts in Service-Oriented Computing (SOC) and be able to provide opinion based on experience, on how best to adapt SBAs. Suitable practitioners were identified through user groups on the Linkedin[4] professional network as well as SOC conference proceedings. After sending out invitations to over 100 practitioners there were 15 replies with seven eventually committing to interviews. At the end of each interview participants were asked to identify others who may participate with some initial positive responses. However, the initial interview partners did not provide leads for additional interviews indicating that chain sampling can be ineffective for such a specific research project topic. Table 2 provides an anonymous profile of the SOC experts that participated in the survey. The company specific details are intentionally vague to protect their identities.

In advance of the interviews the interviewees were sent a copy of the interview guide as well as some background information on the study, allowing them to familiarise themselves with the concepts and terminologies involved in the study. It also gave them extra time to think about the adaptation activities that they encountered while developing SBAs. The interviews were conducted either on site or through the Skype™Voice Over Internet Protocol (VOIP) service which allows the recording of interviews. Once all of the interviews were complete, they were transcribed in preparation for data analysis.

[4] http://www.linkedin.com.

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

Table 2
Survey participants.

Participant	1	2	3	4
Role	Security R&D	Business analyst	Sales/product development	Service composition R&D, commercialising research
Experience with adaptable SBAs	Yes	Yes	No	Yes
Location	USA	Mexico	UK	Germany
Company	A	B	C	D
Size	Very large multinational	Large	Large	Very large multinational
Sector	Hardware/software development	Banking	Mechanical engineering	Communications

Participant	5	6	7
Role	Performance testing	Consultant	Business process improvement
Experience with adaptable SBAs	No	No	Yes
Location	Germany	Netherlands	USA
Company	E	F	G
Size	Very large multinational	Very large multinational	Very large multinational
Sector	Industrial, healthcare	Software development	Software development/ services

Table 3
Case study participants.

Participant	8	9	10
Role	Chief technology officer	Business analyst	Developer
Experience with adaptable SBAs	Yes	Yes	Yes
Location	UK	UK	UK
Company	H	H	H
Size	Medium	Medium	Medium
Sector	Software development	Software development	Software development

3.2.2. Case study

A single case study was also conducted in order to supplement the survey conducted during this inquiry. The case study was based on a typical company with expertise in developing SBAs. The aim of this study [27] was to identify how a SBA development team approaches the adaptation of SBAs. The company where the case study was carried out will be referred to as SbaSoft.

Interviews were used in order to collect qualitative data during the case study. Three stakeholders involved in different aspects of the application development life-cycle were interviewed: the Chief Technology Officer (CTO), a business analyst and a developer. The interviews for the case study were conducted on site at SbaSoft. Table 3 provides an anonymous profile of the case study participants. Again, the company specific details are intentionally vague to protect its identity.

The interviews for the case study were based on a semi-structured interviews similar to that used in Section 3.2.1. However, rather than focusing on adaptation specific activities, the questions were aimed at documenting SbaSoft's entire development life-cycle including adaptation specific activities. Participants in the case study were asked questions such as:

1. Does your company use service based applications, provide services for internal use or third parties: i.e. service client, service provider, or both.
2. Can you briefly outline how requirements are gathered for your services?
3. Can you briefly outline the process followed for designing web services in your company (i.e. people involved, activities carried out)?
4. Are your services designed to be adaptable?

Their entire development life-cycle was documented in order to view their adaptation specific activities in context. After the interviews were complete they were analyzed and provided input for the model.

3.2.3. Data analysis

Once the data collection from the survey and case study was complete the interview transcripts were analyzed using data analysis techniques as described in Miles and Huberman's [28] expanded source book. The raw data was first open coded with descriptive codes being added to segments of the data. A starting list of codes was based on the frame of reference model constructed in the first half of this study. Starting codes, as illustrated in Table 4, were created for activities, stakeholders and artifacts related to SOAdapt-FoR's high-level processes. New codes were added and existing ones modified to accurately reflect the raw data as the coding proceeded. Participant background information was coded in chunks containing information about their current roles as well experience which provides context for their interview responses.

The coding was conducted with ulQda [29], a LaTeX[5] [5] qualitative data analysis package. This package allows codes to be embedded within the raw data where they can be easily reviewed and edited in context. The package also allows the raw data to be typeset to a high quality document, and annotated with the appropriate codes. Finally, this package allows the codes and their corresponding raw data to be exported to a Comma Separated Value (CSV) file. CSV files are easily imported into spreadsheet applications for subsequent

[5] http://www.latex-project.org/.

Table 4
Starting codes.

Category	Process	Code
Activity	Requirements engineering	activity!requirements engineering
	Design	activity!design
	Construction	activity!construction
	Deployment and provisioning	activity!deployment and provisioning
	Operation and management	activity!operation and management
	Adaptation triggers	activity!adaptation triggers
	Adaptation strategy	activity!adaptation strategy
	Enact adaptation	activity!enact adaptation
Artifacts	Requirements engineering	artifacts!requirements engineering
	Design	artifacts!design
	Construction	artifacts!construction
	Deployment and provisioning	artifacts!deployment and provisioning
	Operation and management	artifacts!operation and management
	Adaptation triggers	artifacts!adaptation triggers
	Adaptation strategy	artifacts!adaptation strategy
	Enact adaptation	artifacts!enact adaptation
Stakeholders	Requirements engineering	stakeholders!requirements engineering
	Design	stakeholders!design
	Construction	stakeholders!construction
	Deployment and provisioning	stakeholders!deployment and provisioning
	Operation and management	stakeholders!operation and management
	Adaptation triggers	stakeholders!adaptation triggers
	Adaptation strategy	stakeholders!adaptation strategy
	Enact adaptation	stakeholders!enact adaptation

analysis.her than being described by a single code, each passage was broken down into 5 categories of: processes, activities, tasks, stakeholders and artifacts. Processes, activities and tasks, in this context, relate the the process constructs defined in ISO/IEC 15288 where each process is made up of activities which are in turn made up of tasks. For example, the following passage was assigned the code "activity!operation and management!monitoring" during open coding:

> *I would monitor the state of the task, the external events, also the state of the information, and the documents and the things that are exchanged*

During the refinement process, a high-level process, an activity and 3 tasks were identified, with no stakeholders or artifacts being identified. The process constructs identified during this example are shown in Table 5.

In another example the following quotation provided the activity "Adaptation strategy based on system learning":

> *. . .it would be great to have some type of learning system, in order to be able to learn from the jobs that have been run, failure modes and then to adapt the system according to what it has learnt, and any knowledge we have put in as to why the jobs might fail.*

Once these process attributes were identified they were organized into a conceptually clustered matrix [28]. Similar concepts, in this case the process attributes, were grouped together in to a matrix with activities, stakeholders and artifacts assigned to the appropriate high-level life-cycle processes. They were then ordered into a logical process hierarchy making it is easier to identify duplicate or disjointed attributes. Duplicate attributes were eliminated, while disjointed attributes were reevaluated to see where they might fit into other processes or activities. Finally, the SOAdapt-FoR model was updated to reflect the processes, activities, tasks, stakeholders and artifacts identified during the data analysis. As well as additions, SOAdapt excluded some of the original SOAdapt-FoR constructs to reflect the findings of the empirical inquiry. All of the stakeholders and artifacts from SOAdapt have come from the empirical inquiry since there were none identified in SOAdapt-FoR. A graphical representation of SOAdapt was constructed using the Business Process Modeling Notation (BPMN) notation, showing the processes and activities. This approach was chosen because there were 38 stakeholders, 35 tasks and 17 artifacts identified which would clutter a complete BPMN representation. Tables were constructed to show the complete set of process attributes for each of the processes in the process model.

3.2.4. Evaluation

The model was evaluated to get an impression of its external validity. Several studies were chosen from the literature to be used during this evaluation. These studies were not used in the construction of the reference model since they could not later be used for an impartial evaluation. Oreizy et al.'s process model [30,21] was chosen for the evaluation rather than construction since component-oriented development is related rather than part of service-oriented development. Durvasula et al.'s life-cycle [31] was reserved for evaluation since it could be used to evaluate each of the attributes of our model during a development life-cycle scenario.

To show that the process model is suitable for adaptation it was also compared to a related process model proposed by Oreizy et al. [30,21] for component-based software adaptation. Oreizy et al.'s process model was validated through a worked example where it is used to adapt a cargo routing application. The objective of this comparison is to show that SOAdapt is suitable for runtime adaptation in general but not necessarily component-based adaptation. Therefore, the comparison will be at a high-level of granularity without much emphasis on the lower-level component or service-based entities. If this comparison shows that SOAdapt is useful for adaptation in general, it remains to be seen whether SOAdapt is useful in a service-oriented environment. This will be investigated in the next part of the evaluation.

To demonstrate the transferability of the process model to a service-oriented environment, an evaluation was conducted in order to determine if it is compatible with a SBA development life-cycle proposed by Durvasula et al. [31]. Durvasula et al.'s life-cycle was collaboratively developed by 10 SBA development practitioners from 8 different companies. During the evaluation, each of the activities from the adaptation process model was mapped to life-cycle processes within Durvasula et al. life-cycle. Mapping the activities to the life-cycle without conflict, shows that the process

Table 5
Data analysis example.

Code	Text	Process	Activity	Task	Stakeholder	Artifact
Activity!oam!monitoring	I would monitor the state of the task, the external events, also the state of the information, and the documents and the things that are exchanged	Operation and management	Monitoring	Monitor processes		
		Operation and management	Monitoring	Monitor events		
		Operation and management	Monitoring	Monitor data		

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

model is suitable for use with an empirically based SBA development life-cycle.

4. Company backgrounds

All of the companies that participated in this study have successfully developed adaptable SBAs or frameworks that facilitate the development of adaptable SBAs. These applications consisted of Commercial Off-The-Shelf (COTS) products or bespoke customer solutions. Each of the applications have strengths and weaknesses and each have been developed using different development processes. By reviewing the type of applications developed by the participant companies, we can judge the relevance of the participants to the project. Here are some examples of those applications:

Company A provide a service monitoring COTS framework which facilitates SBA adaptation. The framework monitors application events that may trigger adaptation. These events are also permanently stored for later investigation or troubleshooting. Here is an excerpt from Participant 1's description of the framework:

> . . .*and* it will actually store all of these audit events that come in there and so later even the year two years later you can see was I compliant with this particular control objective during this particular time window two years ago.

On the other hand Company B have implemented a bespoke adaptation solution to facilitate adaptation in their in house enterprise application suite. This solution consists of a service broker that intercepts service and re-routes their path depending on their properties. Here is a description of the application:

> . . .*on one of the systems I worked on there were rules that allowed some adaptation, and we had to interview financial users that they knew the rules, so for example they were saying that if you have a message of number type 100 in one filed, and one field was filled with some value then this message has to be sent to this office if not it has to be sent to some to some other office. But sometimes you have to decide when you receive the message where to send it, so there was a kind of automatic broker.*

Company 4 like Company 1 also provide a COTS adaptation framework, the application focused on compositional adaptation. A problem with this framework is that there is no runtime monitoring or governance which will rescue the application during an expected event, so all adaptation configurations are tested thoroughly during design time.

> *With our mechanism you can use in run time so the adaptation process itself can enforce boundaries but there's no, there's no multi level that will somehow, you know, play some kind of arbitrator role and stop the execution if something goes wrong. But that's exactly the reason why it's so important to actually be able to test in advance and guarantee that nothing will actually go wrong.*

Company H where the case study was conducted was another COTS development company. In this case they developed user interface software for large enterprise customers. Their application allow the consumption of data services that could be merged into collated views. Their framework allows their clients to adapt their SBAs through "drag and drop" admin interfaces. Here is an excerpt from a description of the framework:

> . . .*yeah* customers can do it if they got like some, you know a degree of technical ability, because it is just basically a drag and drop environment, so yeah we do that, quite a lot of customers have wanted to do it for themselves so they get training courses from us.

5. SOAdapt-FoR model

Ten generic adaptation activities were distilled from a set of 50 activities that were identified in from Lane and Richardson's SLR [7] as well as other related literature. The SLR included a methodical search of computer science related digital libraries such as IEEE Xplore and the Web of Science. In total 722 studies were identified in the initial search with 77 found to contain process details for developing SBAs. These SBA development processes were then investigated to see whether or no they support the development of adaptable SBAs. A total of 8 studies from the SLR were found to have supported adaptation. These 8 studies along with two others found in the literature are listed in Appendix A.

The 10 generic activities and their parent processes identified in the S-Cube life-cycle were ordered into a sequential workflow which shows how each activity relates to one another creating a process model. BPMN is used to visualise the model. BPMN has a standard modeling notation which should be understandable by a large number of business as well as technical stakeholders [32]. Fig. 3 illustrates the SOAdapt-FoR model represented in BPMN in contrast with the block diagram used to construct the S-Cube life-cycle.

5.1. SOAdapt-FoR model description

Our description of the SOAdapt-FoR model pays particular attention to the activities and their interrelations. The high level process such as "Requirements Engineering" and "Design" that make up the skeleton of the model are primarily inherited from the S-Cube life cycle model with some exceptions. The "Requirements Engineering and Design" process inherited from the S-Cube life-cycle model was separated into two separate processes to accommodate the numerous activities identified for each process.

The S-Cube life cycle is an amended version of Papazoglou and Van Den Heuvel's [2] service development life cycle with an adaptation cycle added to it's left hand side [18]. Papazoglou and Van Den Heuvel's original life-cycle was influenced by approaches such as the Rational Unified Process (RUP) [1], Business Process Modelling (BPM) [33] and Component-Based Development (CBD) [19].

It is important to reiterate at this stage that the SOAdapt-FoR model is not a complete life-cycle model as it only contains activities that are related to adaptation. These activities are mapped to the S-Cube life-cycle model which indicates where they should occur during the software development life-cycle. The activities could also be mapped to similar processes within other development life-cycles with similar high level process of the S-Cube life cycle.

The frontier between SOAdapt-FoR and its inclusion in an existing life-cycle models depends on the type of model it is being integrated with. In the evolution cycle, adaptation activities should occur before the other activities in the process have been completed. This is necessary so that other activities can see the impact of the activities from SOAdapt-FoR. For example, adaptation may require a particular service granularity and service hierarchy and these aspects need to be decided before the remainder of the application is developed. The activities from the adaptation cycle (left-hand side) are contained in complete processes. Therefore, these processes may occur autonomously during the operation of the application regardless of the application's development life-cycle.

If SOAdapt-FoR was used in a development project employing a waterfall development method, each of the process groups in the evolution cycle would have to be mapped to their equivalent process in the waterfall model. This is a straightforward example as each of the procesess from the evolution cycle of SOAdapt-FoR usually have an equivalent in most waterfall model instantiations. The

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

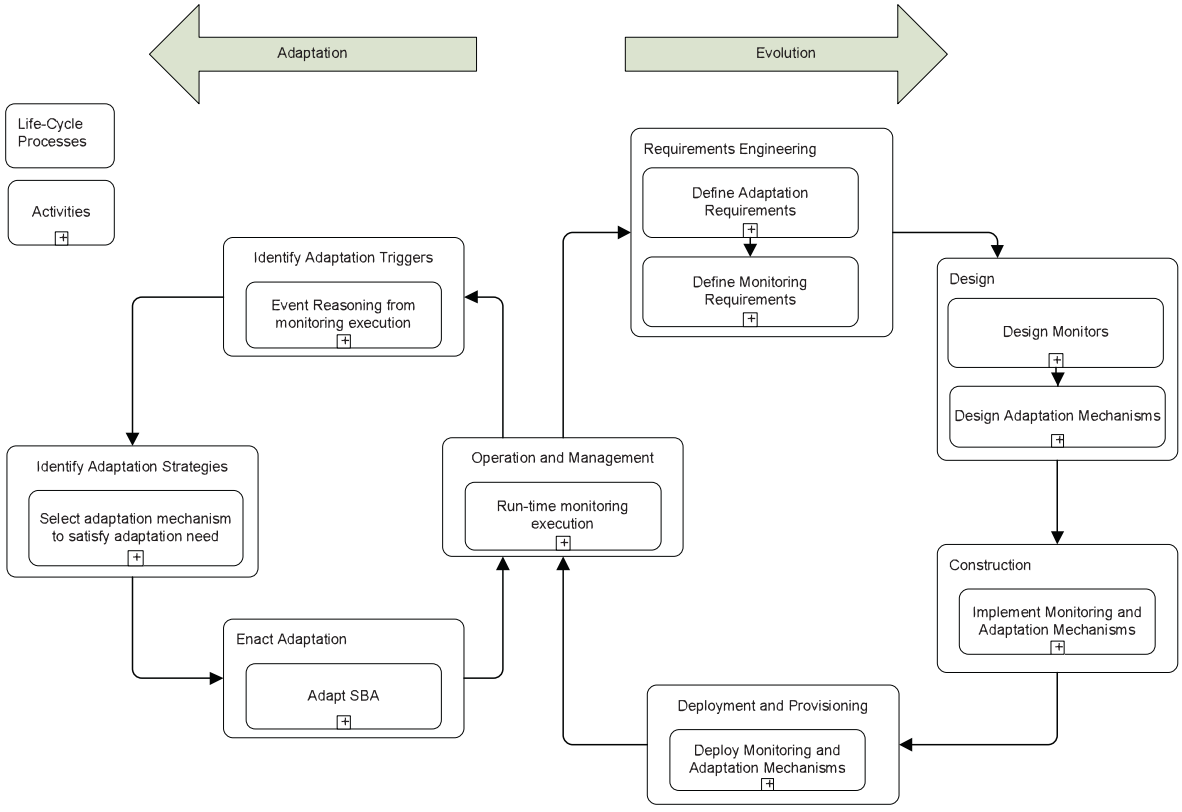

Fig. 3. SOAdapt-FoR model.

adaptation cycle of SOAdapt-FoR would then need to occur in series with the other processes from the waterfall. This would result in a hybrid linear/iterative development life-cycle not to dissimilar to the SCRUM development life-cycle. If was mapped to an existing SCRUM model the result would be a hybrid linear/iterative life-cycle with an iterative development cycle and an iterative adaptation cycle in series with the other linear SCRUM processes.

5.1.1. Requirements engineering

In the SOAdapt-FoR model we are interested in the capturing adaptation specific requirements for SBAs. Adaptable SBAs will also have many other requirements that are not related to adaptation, but those are outside the scope of the SOAdapt-FoR model. Within SOAdapt-FoR there are two requirements engineering activities that are specific to adaptation:

Define adaptation requirements refers to the activity of defining or eliciting requirements relating to why it is necessary for the SBA to adapt. An adaptation requirement may be, for example, if the Quality of Service (QoS) of service A falls below a minimum threshold then the application adapts by choosing an alternative service. The adaptation requirement may also state how the adaptation occurs, adaptation might simply involve choosing between two hard coded service end points or in a more complex way by negotiating with a service directory. An adaptation requirement may also specify whether or not the adaptation occurs with user intervention or automatically with minimal or no user input.

Define monitoring requirements Once adaptation requirements have been determined, monitoring requirements need to be defined in order to monitor service attributes that relate to adaptation requirements. For example, in order to enable an application to adapt based on QoS, it is necessary to monitor relevant service quality attributes such as service latency and service reliability. So, in this case, the *monitoring of service latency and reliability* is a monitoring requirement, relating to the *adapt if service QoS falls below a minimum threshold* adaptation requirement.

5.1.2. Design

There are two adaptation specific design activities within the design process of SOAdapt-FoR:

Design monitors is a design activity where monitors are designed in order to satisfy the monitoring requirements defined during the requirements engineering process. Details such as implementation technology and infrastructure need to be considered at this stage. The monitor designs need to be included with a design specification document or a similar alternative that can be used by the application programmers.

Design adaptation mechanisms. Once monitors have been designed, a means to adapt the application based on monitored events must be designed. The adaptation mechanism must have the ability to respond to monitored events and adapt by enacting an adaptation strategy. Bucchiarone et al. [24] identified many possible adaptation strategies such as substituting services with known alternatives or service re-negotiation which involves negotiation with a service directory in order to locate alternative services. It is at this point that it must be determined which of these strategies will be used in order to adapt during runtime. Once adaptation strategies have been decided upon the technical and infrastructural details of how they are implemented and triggered need to be designed.

5.1.3. Construction

The adaptation specific construction activity included in SOAdapt-FoR is outlined below:

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

Implement monitoring and adaptation mechanisms. This activity involves the implementation or coding of the monitoring and adaptation mechanisms that were discussed in Section 5.1.2. This activity would occur along with all of the other application implementation activities within the software development project. There are no large design features to be decided at this point as they are specified during the design process. However, due to the complex nature of adaptation mechanisms, this implementation in one of the most challenging parts of the development process.

5.1.4. Deployment and provisioning

The S-Cube life-cycle has inherited its deployment and provisioning process from Papazoglou and Van Den Heuvel's [2] service development methodology, which itself in influenced by the RUP methodology. The process specifies how a software system is deployed as well as provisioned or metered, which would allow charging for a service. Here is an outline of the single, adaptation specific, deployment and provisioning activity from the SOAdapt-FoR model:

Deploy monitoring and adaptation mechanisms. The deployment of monitoring and adaptation mechanisms are most likely going to get deployed with the other parts of the SBA and may not require any special treatment. However, if the monitoring or the adaptation mechanisms are built into a service infrastructure which is external to the application they will have different deployment requirements.

5.1.5. Operation and management

The operation and management of the SBA is one of the most critical processes with respect to adaptation. Whether its manual or automatic, its during the operation of the application when certain monitored events prompt the adaptation of the application. Within the SOAdapt-FoR model and the S-Cube life-cycle, the operation and management process is the link between the evolution cycle which undergoes typical maintenance and the adaptation cycle which allows the application to adapt. As well as providing the link between the adaptation and evolution cycles it also has its own activities, and one in particular that relates to adaptation:

Run-time monitoring execution. Run-time monitoring is a key activity within the operation and management process; it monitors events that signal whether or not run time adaptation should take place. This could be a manual activity or fully automatic depending on the application, but most likely it would be necessary to have some form of involvement by a relevant stakeholder.

5.1.6. Identify adaptation triggers

Identify adaptation triggers is a process which makes use of monitoring mechanisms in order to identify reasons to adapt. This process occurs during the adaptation cycle, thus all of its sub-activities are related to adaptation. Within SOAdapt-FoR there is one activity in this process which will now be explained:

Event reasoning from monitoring execution is an activity that involves the identification of monitored events from application monitors. These events may be reasoned or identified manually depending the complexity of the application. In an fully automated system events would be identified based on application logic, however, in a manual scenario, events might be manually identified by end users or developers through inspection of application logs.

5.1.7. Identify adaptation strategies

During this process, adaptation strategies are selected to suit the monitoring events that were identified in the previous process. Different event types may be associated with different adaptation strategies, and again, this association may be built into the applications logic may be made at runtime by a stakeholder. One scenario is that an application has many built in adaptation strategies where the most appropriate one is selected by an end user at runtime.

Select adaptation mechanism to satisfy adaptation need. This activity is the only activity within the identify adaptation strategies process. This activity completes all of the goals of its parent process which is to select and adaptation mechanism which is most suited to the monitored event that triggered the adaptation. The specifics actions within this activity are difficult to determine with out having some knowledge about the adaptation engine used in the underlying application.

5.1.8. Enact adaptation

This process is the last process in the adaptation cycle which simply involves the enactment of the adaptation strategy that was selected. Within the context of SOAdapt-FoR, this process has only one activity which is to adapt the SBA.

Adapt SBA. This activity refers to physically adapting the application. If it is an autonomous application this activity is automatic. If the application is not autonomous enacting the adaptation of the SBA may require configurations modifications by end users or developers.

6. SOAdapt

In this section we will present the results of the content analysis that carried out on the 10 interview transcripts (275 min of audio) conducted during this study. Once the content analysis was complete the SOAdapt-FoR model was updated to represent the study's findings. Fig. 4 is a BPMN representation of the empirically grounded process model. The illustration shows high-level life-cycle processes and activities. Tasks, stakeholders and artifacts are listed separately in the following sub-sections.

6.1. Processes and activities

Interview participants agreed that the high-level processes from the SOAdapt-FoR model should remain largely unchanged and that adaptation activities should occur in the traditional requirements engineering, design, construction, deployment and provisioning and operation and management phases. However, a need for a separate testing process was identified. Another notable deviation from the original high-level processes was the lack of emphasis placed on the provisioning sub-process from the deployment and provisioning process.

The processes from the adaptation cycle of the SOAdapt-FoR model caused confusion among some participants during the interviews. There was confusion between terms such as strategies, mechanisms and their applications. These processes as well as their contained activities have now been modified to make them more intuitive.

Changes to the activities contained within the high-level processes were more extensive than changes to their parent processes. This is to be expected given that the high-level life-cycle processes from the evolution cycle are almost De facto, featuring in prominent life-cycle models such as RUP, ISO/IEC 12207 and

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

the waterfall model [34]. All of the changes made to the processes and activities from the SOAdapt-FoR are listed in Table 6. The first column of the table shows whether the processes or activities were added or removed. In most of the cases activities were modified rather than being removed completely. This is shown in the table as a *remove* followed by an *add* of the modified version. Rows with both a *process* and *activity* entry refer to the activity and show the process for reference, while rows with just a process entry refer directly to the process. In cases where there is no directly supporting data, changes were made to activities so that they correspond to the activities that have supporting data. For example, the modification of "Deploy monitoring and adaptation mechanisms" to "Deploy adaptation strategies and monitors" did not have directly supporting evidence. This activity was changed in order to correspond with the activity "Design adaptation strategies" which did have supporting data. The "Provisioning" sub-process was removed from the "Deployment and provisioning" process, not as a result of comments from the interviewees to remove it, but because when probed, they never offered any details on adaptation activities that need to occur during provisioning.

6.2. Detailed process attributes

The following subsections provide an outline of the activities in each of the process groups from SOAdapt. The activities and tasks in Tables 7–15 are color coded depending on whether or not they can be modified to meet the needs of a particular project scenario. The main flexibility points relate to whether or not the type of adaptation chosen for the project will be static (adaptation paths determined at design time) or dynamic (adaptation paths chosen at runtime).

6.2.1. Requirements engineering

The two Requirements Engineering (RE) activities from the SOAdapt-FoR model remain unchanged in SOAdapt, with three tasks having being identified for the "Define Adaptation Requirements" activity and 2 for "Define monitoring requirements" (see Table 7). There were several types of adaptation triggers identified for "Define events that trigger adaptation" task. For example, learning-based, context-based, availability based, rule-based and QoS Based triggers were identified. The availability and QoS based adaptation trigger was elicited form the following quotation:

> . . .*so* if one of the systems go down, to be able to say, hey one of the systems has gone down, identify what are possible reasons for it to go down, and then make decisions based on the information that you've got as to whether to retry it on that system, retry it on another system, go back to the customer, or what ever, thats the kind of think that we were looking at.

These adaptation activities and tasks are self contained and can be carried out in parallel with the other RE tasks that are necessary for a project. A number of adaptation related artifacts and stakeholders were also identified for the process.

Unlike the activities and tasks, the artifacts and stakeholders identified are not exclusively concerned with adaptation. For example, all of the adaptation stakeholders listed here usually have other roles in other RE activities.

6.2.2. Design

There was one prominent activity change between the SOAdapt-FoR and SOAdapt for the design process. The "Design Adaptation Mechanisms" activity was changed to "Design Adaptation Strategies". Interviewees felt that Mechanisms was too ambiguous to describe the was in which SBAs adapt:

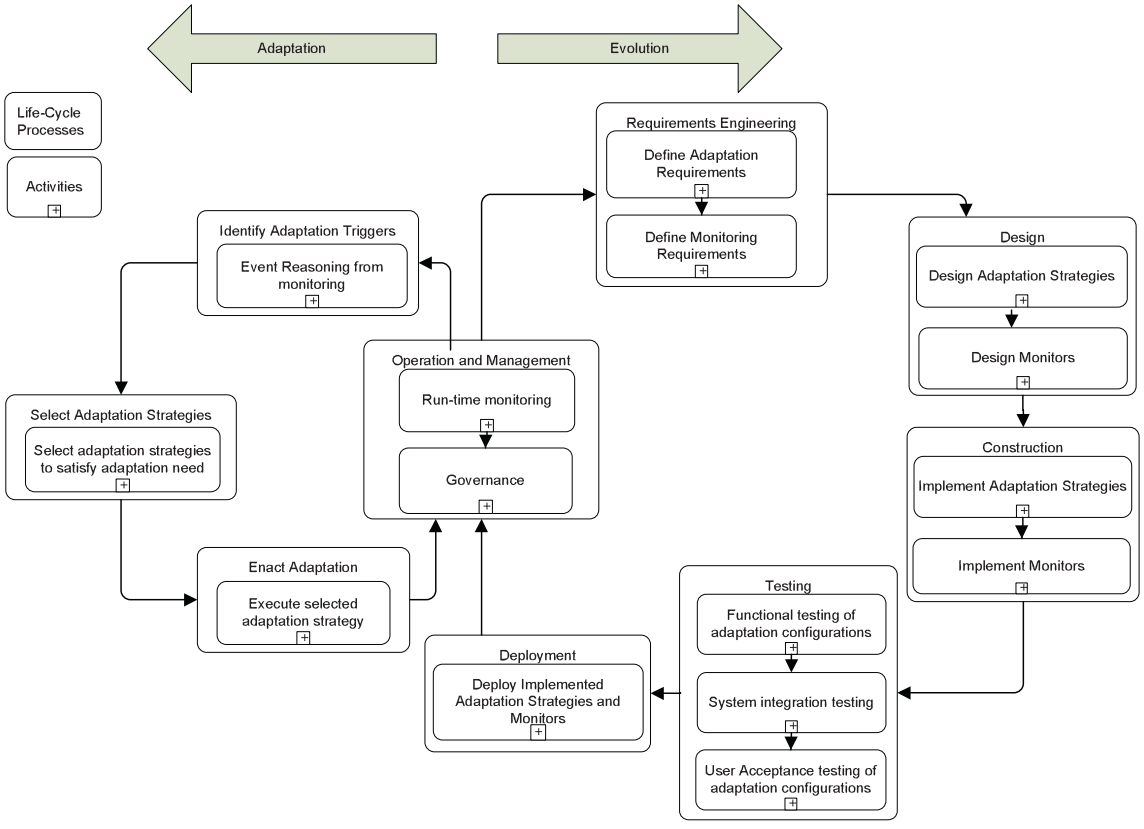

Fig. 4. SOAdapt – process model for adapting SBAs.

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

Table 6
SOAdapt process and activity changes.

Add/ remove	Process	Activity
Remove	Design	Design adaptation mechanisms
Add	Design	Design adaptation strategies
Remove	Construction	Implement monitoring and adaptation mechanisms
Add	Construction	Implement adaptation strategies
Add	Construction	Implement monitors
Remove	Provisioning	
Remove	Deployment	Deploy monitoring and adaptation mechanisms
Add	Deployment	Deploy adaptation strategies and monitors
Remove	Operation and management	Run-time monitoring execution
Add	Operation and management	Run-time monitoring
Remove	Identify adaptation triggers	Event reasoning from monitoring execution
Add	Identify adaptation triggers	Event reasoning from monitoring
Remove	Identify Adaptation strategies	
Add	Select adaptation strategies	
Remove	Select adaptation strategies	Select adaptation mechanisms to satisfy adaptation need
Add	Select adaptation strategies	Select adaptation strategies to satisfy adaptation need
Remove	Enact adaptation	Adapt SBA
Add	Enact adaptation	Execute selected adaptation strategy
Add	Testing	

"Yeah, so I would call these three things out on the right side that I design my monitor, my strategy and my adaptation. I do those three things because my monitor leads to correct strategy".

Adaptation strategies such as service discovery and service substitution were also identified during the inquiry. There were eight adaptation tasks identified for designing these adaptation strategies, along with two tasks for designing monitors (see Table 8). There were eight adaptation artifacts and four adaptation stakeholders also identified for this process. The artifacts and stakeholders identified, like the RE process, are not exclusive to adaptation related activities. In this process artifacts such as "Functional Design" documents are contributed to be adaptation activities, but are also contributed to be non-adaptation activities.

6.2.3. Construction

The construction activity "Implement monitoring and adaptation mechanisms" was updated to "Implement adaptation strategies" to reflect the new design activity "Design adaptation strategies". The "Implement Monitors" activity was isolated as a separate activity to show the "Construct monitors" task in its specific context. A noteworthy task identified for the implementation process was "Early Testing" with the other tasks specifically related to construction. The complete adaptation process attribute details for the construction process, including three artifacts and two stakeholders, can be seen in Table 9.

6.2.4. Testing

The testing process was added to SOAdapt as a result of three testing activities that were identified by interviewees. Along with the testing process there were adaptation related testing activities identified for the "Design", "Construction" and "Deployment" processes. Since the testing activities were identified from scratch, rather than being derived from SOAdapt-FoR activities, the interviews did not provide a lot of detail about tasks, artifacts and stakeholders. The activities and single task identified are illustrated in Fig. 10.

Table 7
Adaptation process attributes for the requirements engineering process.

Requirements engineering
Adaptation activities and tasks
Define adaptation requirements:
– Define events that trigger adaptation (learning-based,context-based, availability based, rule-based, QoS Based)
– Choose between manual and automatic adaptation
– Validate adaptation requirements
Define monitoring requirements:
– Define application components that have to be monitored
– Define how monitored events are reported

Adaptation artifacts
Adaptation workflow
Use cases
Architectural overview

Adaptation stakeholders
Business analyst
Business stakeholders
Solution architects
Engineers
Developers
Information security team
System administrators
End users

Table 8
Adaptation Process Attributes for the Design Process.

Design
Adaptation activities and tasks
Design adaptation strategies:
– Design adaptation strategies (service discovery, service substitution)
– Define adaptation workflows
– Design adaptation dashboard (manual adaptation)
– Design logic to match monitored events to adaptation strategies (automatic adaptation)
– Design service interfaces
– Choose service alternatives
– Use guide for selecting services
– Adaptation strategies prototyping
– Design tests for adaptation strategies
– Design monitoring services (post-mortem, real time monitoring)
– Determine performance impact of monitors
Design monitors:
– Design monitoring services (post-mortem, real time monitoring)
– Determine performance impact of monitors

Adaptation artifacts
Service topology
Interface definitions
Input/output data
Software specifications
Functional design
Enterprise architecture

Adaptation stakeholders
Requirements engineers
Solution architects
Developers
Maintainers

6.2.5. Deployment

The "Deployment" process originated as the "Deployment and Provisioning" process from the SOAdapt-FoR model. As previously mentioned (Section 6.1) the "Provisioning" process was dropped because interviewees did not mention it in relation to adaptation. The process contains one adaptation activity: "Deploy Adaptation Strategies and Monitors" which was changed from

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

Table 9
Adaptation process attributes for the construction process.

Construction
Adaptation activities and tasks
Implement adaptation strategies:
– Construct adaptation strategies
– Implement adaptation dashboards
– Early testing
– Refinement
Implement monitors:
– Construct monitors
Adaptation artifacts
Middleware
Technical design
Dashboards
Adaptation stakeholders
Engineers
Maintainers

Table 10
Adaptation process attributes for the testing process.

Testing
Adaptation activities and tasks
Functional testing of adaptation configurations:
– Prove to customer that application will work in all adaptation cases
System integration testing
User acceptance testing of adaptation configurations
Adaptation artifacts
Adaptation stakeholders

Table 11
Adaptation process attributes for the deployment process.

Deployment
Adaptation activities and tasks
Deploy adaptation strategies and monitors:
– Deploy components on separate systems
– Monitor system dependancies
– Functional testing
Adaptation artifacts
Adaptation stakeholders
Developers
Maintainers
Service providers
Service consumers

Table 12
Adaptation process attributes for the operation and management process.

Operation and management
Adaptation activities and tasks
Run-time monitoring:
– Audit service delivery infrastructure
– Notify users of critical events
– Monitor processes
– Monitor events
– Monitor data
Governance:
– Govern adaptation
– Negociate SLAs
Adaptation artifacts
State model
Adaptation stakeholders

Table 13
Process attributes for the identify adaptation triggers process.

Identify adaptation triggers
Adaptation activities and tasks
Event reasoning from monitoring
Adaptation artifacts
Adaptation stakeholders

Table 14
Process attributes for the select adaptation strategies process.

Select adaptation strategies
Adaptation activities and tasks
Select adaptation strategies to satisfy adaptation need:
– Select adaptation strategy based on recommendation
– Select adaptation strategy from an end used adaptation dashboard
– Select adaptation strategy to satisfy adaptation need
Adaptation artifacts
Adaptation stakeholders
End user

Table 15
Process attributes for the enact adaptation process.

Enact adaptation
Adaptation activities and tasks
Execute selected adaptation strategy
Adaptation artifacts
Adaptation stakeholders
End user

"Deploy Monitoring and Adaptation Mechanisms" in order to match the terminology of preceding processes. The deployment of monitors and strategies were kept as a single activity as the deployment tasks apply generally to both. As illustrated in Table 11, there were 0 adaptation artifacts and four adaptation stakeholders identified.

6.2.6. Operation and management

The Operation and Management process has retained its monitoring "Run-time monitoring", from the SOAdapt-FoR model, and has gained an activity called "Governance". The governance activity, as the name implies, contains tasks for governing how the SBA adapts during run-time. The "Run-time monitoring" activity has been updated with tasks for monitoring data, events and processes as well as a user notification task. The "Governance" activity has tasks for the governing the adaptation as well as negotiating Service-Level Agreements (SLAs). A "State model" was the only adaptation artifact identified without any relevant stakeholders being identified (see Table 12). A state model is an important artifact in the operation of autonomous adaptable SBAs. The state model is the blue print which tells the application how to respond to certain states or events.

6.2.7. Identify adaptation triggers

The "Identify adaptation triggers" process has not changed in SOAdapt, it still retains the single Activity: "Event Reasoning from monitoring", with no lower level tasks. However, this activity was confirmed as a relevant adaptation task during the interviews:

> *"What are the kind of criteria that we would want for this type of adaptation, the trigger points for doing some adaptation".*

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

As Table 13 illustrates, there were no adaptation artifacts or stakeholders identified during the interview process.

6.2.8. Select adaptation strategies

The "Select adaptation strategies" process, like the "Identify adaptation triggers" process, retains its original activity from the SOAdapt-FoR model. However, in this case the 'Select adaptation strategies" activity has been updated with three tasks for selecting adaptation strategies. A single stakeholder, "End user", has also been identified for the process (see Table 14). End users play an important role in identifying the correct adaptation strategies when the adaptation process is not fully automatic.

6.2.9. Enact adaptation

The final process, "Enact adaptation", has also not had many process attributes identified. The wording of its single activity "Execute selected adaptation strategy" had been reworded to match the terminology of other activities but its intent remains the same. The "End user" stakeholder has also been identified for this process (see Table 15). Again end users have the final say on enacting adaptation when the application is not fully automatic.

7. Evaluation

We employed two methods to determine the external validity of the process model developed during this research. In Section 7.1, we describe the comparison between the process model and a previously validated component-based software adaptation process model, indicating whether the model contains the required activities for adaptation. In Section 7.2, the activities from the process model are mapped to an SBA development life-cycle which demonstrates their transferability.

7.1. Inter-model evaluation

To evaluate SOAdapt's capability of adapting SBAs we compared to a similar model proposed and validated by Oreizy et al. [30,21] for adapting component-based applications. Component-based applications are similar to service-based applications, with both application types being composed of loosely coupled self-contained software modules. This similarity allows comparisons to be made between the two approaches. However, the different development paradigms being addressed by each means that a like for like comparison is not being made.

The two models will be compared at an activity level since Oreizy et al's model does not specify the life-cycle processes within which its activities occur. Each of Oreizy et al.'s activities will be analyzed to see if they have equivalent or comparable activities within SOAdapt. The degree of similarity between the activities will also be judged by degree of similarity between their component tasks. Oreizy et al.'s do not explicitly list the tasks within each activity, however they are described verbally within their paper. These task descriptions have been extracted for the comparative analysis.

7.1.1. High-level comparison

As with SOAdapt, Oreizy et al.'s adaptation model has two cycles: an adaptation cycle and an evolution cycle. Both models implement changes to applications during an offline evolution cycle. However, the overall approaches for adaptation are different. In SOAdapt, adaptation mechanisms are built into the application at runtime. Then changes can be made to the operational application using these mechanisms. Oreizy et al change the composition of the application during evolution, while during adaptation implement the changes enacted during evolution. They argue that making changes during evolution is safer because it allows the application to be tested thoroughly before reflecting those changes in the running system. This is a valid consideration which is addressed in SOAdapt by the task "Prove to customer that application will work in all adaptation cases". This task is completed during the testing process and ensures that possible application configurations are tested off line during evolution before the application is made operational. The differences in the adaptation approach may be rooted in the fact that components are more tightly coupled than services and hence require a greater deal of testing and integrity checks during evolution. The two approaches share many of the same activities. For example, both approaches have runtime monitoring processes to detect when adaptation should be enacted. Both methods also have activities to govern and manage runtime adaptations.

7.1.2. Detailed comparison

Table 16 was constructed in order to get a better picture of whether SOAdapt contains equivalent adaptation activities and tasks to those outlined by Oreizy et al. The left hand column contains the complete list of activities and tasks from Oreizy et al. while the right hand side shows comparable or similar tasks and activities from SOAdapt.

It is evident from the table that all of the adaptation activities from Oreizy et al. have comparable activities in SOAdapt. This shows that, at a high-level at least, SOAdapt contains the basic ingredients for adaptation. The table does not show the adaptation activities identified in SOAdapt which do not have counterparts in Oreizy et al.'s model. SOAdapt goes beyond the basic set of activities for adaptation and considers adaptation more explicitly during evolutionary processes such as design, deployment and testing. There are some activities where Oreizy et al. has more detail than SOAdapt particularly with regard to its monitoring activities. The similarities that can be seen between the two models suggest that SOAdapt is a valid adaptation model. This, however, is based on the assumption that Oreizy et al.'s model is itself valid and that adapting component-based applications is a comparable process to adapting SBAs.

7.2. Transferability

In order to determine the transferability of SOAdapt, each of its activities were mapped to a component-based application development life-cycle. This life-cycle proposed by Durvasula et al. [31], suggests a high-level process as well as best practice guidelines for developing SBAs. The best practice guidelines are given in a non-sequential format, providing practitioners with a menu of best practices that they can select to use in conjunction with the proposed life-cycle process model. If the activities and tasks from SOAdapt can be intuitively mapped to Durvasula et al.'s development life-cycle, this shows the transferability of the model to a real-world development scenario.

Table 17 illustrates Durvasula et al.'s development life-cycle annotated with the activities from SOAdapt which are highlighted in bold. The granularity of development activities from Durvasula et al.'s life-cycle and SOAdapt are somewhat different, with Durvasula et al. embedding some life-cycle processes within others. For example, Deployment is embedded as an activity within the IT Operations process. This is not a problem, but it does have some knock on effects to the mappings contained in the Table 17. Some of the activities from SOAdapt are represented as tasks to satisfy the different level of process granularity used by Durvasula et al.

It is evident from Table 17 that the activities from SOAdapt fit into the SBA development life-cycle indicating their transferability. Changes in granularity have been made in order to make the mapping. However, this is to be expected given the variation of process

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

Table 16
Oreizy et al. vs SOAdapt.

Oreizy et al.	SOAdapt
A1: Maintain concistancy and system integrity	Governance/design adaptation strategies/functional testing of adaptation configurations
T1: Preserve consistant model of application architecture	Prove to customer that application will work in all adaptation cases
T2: Preserve strict correspondence between architectural model and execution implementation	Govern adaptation
T3: Develop architecture evolution manager (AEM) to mediate changes to the AEM	Design adaptation dashboard (manual adaptation)
A2: Enact changes	Implement adaptation strategies
T4: Use an interactive editor to construct architectures and describe modifications	
T5: Use design tools to critique architectures as they are constructed	Early Testing
T6: Use domain specific tools to check for semantic errors	Refinement
T7: Use modification interpreter to interpret change description language scripts as AEM primitives	Construct adaptation strategies
A3: Collect observations	Run-time monitoring
T8: Observe and provide notifications of exceptional events such as resource shortages	Monitor events/notify users of critical events
T9: Dynamic modification of monitors and monitored events	
T10: Model application behavior as patterns of events	
T11: Remote monitoring by human actors	Audit service delivery infrastructure
A4: Evaluate and monitor observations	Event reasoning from monitoring/functional testing of adaptation configurations
T12: Monitor behaviors of the running application and compere them to behavioral requirements	
T13 Determine all possible architectural configurations to use for concistancy checks	Prove to customer that application will work in all adaptation cases
T14: Automatic runtime concistancy checks	
A5: Plan changes	Define adaptation requirements/define monitoring requirements
T15: Plan which observations are required for enacting adaptations	Define events that trigger adaptation (learning-based,context-based,availability based, rule-based, QoS based)
T16: Plan which adaptations to make and when to make them	Define application components that have to be monitored
A6: Deploy change descriptions	Execute selected adaptation strategy
T17: Use AEM to translate change descriptions into specific application modifications	

attribute granularity in SBA development process models such as those reviewed in [7].

It is also necessary to evaluate the degree of transferability between the deliverables or artifacts of each process model. The mapping in Table 18 was constructed in order to see the level of compatibility between the artifacts. The first column lists the artifacts from SOAdapt while the third column shows the artifacts from Durvasula et al.'s life-cycle. The second column shows how the artifacts from SOAdapt would relate to the artifacts from Durvasula et al.'s life-cycle during a development project.

In the requirements engineering, design and construction processes all of the artifacts can be mapped to Durvasula et al.'s life-cycle except for the software specifications and functional design artifacts. These are common artifacts and are often present in software development projects. They may not be mentioned by Durvasula et al. as they focus on service-specific artifacts. The state model artifact from the operation and management process was the only other artifact without a match in the mapping. This shows a high level of interoperability between the artifacts of the two models. Mapping the artifacts from SOAdapt has had no negative impact on the artifacts already present in Durvasula et al.'s life-cycle. The artifacts in Durvasula et al.'s life-cycle focus on development of service-oriented applications in general while the artifacts in SOAdapt are primarily concerned with adaptation. Therefore, the artifacts from SOAdapt interact with those in Durvasula et al.'s life-cycle as specialized artifacts which enhance the overall development process.

8. Discussion

The research presented in this paper has addressed a gap. In the first instance, while there have been many implementation techniques proposed for adapting services and SBAs, they are not process focused. Secondly, studies have not been based on the experiences and opinions of expert SBA development practitioners. When developing a process model for adapting SBAs it is important to look at the broader perspective of the entire SBA development life-cycle, taking into account the various actors and stakeholders involved. By analysing interviews carried out with developers, analysts, architects, and business stakeholders from various domains, a useful real-world SBA adaptation process model has been constructed.

The approach taken in this paper has leveraged the existing literature, which was used to construct a conceptual framework to guide the fieldwork and qualitative data analysis. There are arguments against this approach, with some suggesting [35] that the use of a conceptual framework influences research participants and results in biased findings. Miles and Huberman [28], however, argue that all qualitative research is inherently biased by the researcher and that the benefits of using a conceptual framework outweigh the drawbacks. In completing this study, we found that participants offered insights that they might not have thought about in the absence of prompts from the conceptual framework.

The data analysis conducted followed a best practice positivistic approach which allows the results to be audited and reviewed by secondary researchers. This rigorous, repeatable approach aims to eliminate researcher bias and enhance the credibility of the research. Furthermore, this approach allows for the findings to be extended by conducting additional interviews and analysing the additional data with the original analysis method.

It was discovered during the interview process that participants were able to readily relate to the processes in the evolution cycle and to provide relevant insights into the adaptation activities that should occur during these phases. This is likely to be due to the fact

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

Table 17
Mapping SOAdapt to Durvasula et al.'s SBA life-cycle.

Life-cycle process	Activity	Tasks
Requirements and analysis	Map high-level business processes	
	Prioritise business services	
	Capture business service requirements	
	Define adaptation requirements	
	Define monitoring requirements	
	Architecture review	Review requirements
		Review alternatives and estimate effort
		Propose solution
	Prioritise and add to solutions portfolio	
Design and development	Assign resources to solutions development team	
	Design solutions – identify reuse opportunity	
	Design adaptation strategies	
	Design monitors	
	Develop, QA and conduct UAT for business solution	Implement adaptation strategies
		Implement monitors
		Functional testing of adaptation configurations
		System integration testing
		User acceptance testing of adaptation configurations
IT operations	Assign resources to service operations teams	
	Identify infrastructure needs and establish systems environment	
	Deploy business solution	Deploy adaptation strategies and monitors
	Maintain solution to business requirements	Run-time monitoring
		Governance
	Identify adaptation triggers	Event reasoning from monitoring
	Select adaptation strategies	Select adaptation strategies to satisfy adaptation need
	Enact adaptation	Execute selected adaptation strategy

Table 18
Mapping SOAdapt's artifacts to Durvasula et al.'s artifacts.

SOAdapt artifact	Relationship	Durvasula artifact
Requirements engineering		
Adaptation workflow	is integrated into	Design models: UML, SCA service assembly model and others
Use cases	is integrated into	Design models: UML, SCA service assembly model and others
Architectural overview	uses	Bindings: JMS, RMI, IIOP, HTTP (s), and others
Design		
Service topology	is integrated into	Design models: UML, SCA service assembly model and others
Interface definitions	is integrated into	Design models: UML, SCA service assembly model and others
Input/output data	is integrated into	Design models: UML, SCA service assembly model and others
Software specifications		
Functional design		
Enterprise architecture	uses	Bindings: JMS, RMI, IIOP, HTTP (s), and others
Construction		
Middleware	uses	Product specific metadata for configuration as well as service execution
Technical design	is used to guide development of	Source code, Jave documents, release notes
Dashboards	form part of	Source code
Operation and management		
State model		

that these life-cycle processes are present in many of the formal development life-cycles that are reported in the literature. Respondents found it more difficult to relate to the processes introduced in the adaptation cycle, which resulted in significantly less data being collected for this these processes. Another interesting finding was that many practitioners did not consider the use of a service specific life-cycle. They normally develop SBAs using traditional waterfall style methods. However, on reviewing the frame of reference model with an additional adaptation cycle they considered this to be a useful approach.

8.1. Threats to validity

The primary threat to the validity of this research is the generalisability of the adaptation model constructed. Every effort was made to select interview partners from a diverse set of companies and a diverse set of roles, but this does not guarantee that the results are generally applicable. In order to mitigate these risks the process mode was mapped to a SBA development life-cycle that was empirically developed independently of this research. The goodness of fit that was achieved during the mapping helps to show the model can be generally applied.

In the absence of having used the model in a real-life scenario it is difficult to determine its capability of adapting SBAs. There are two conceivable methods of determining its real world adaptation capabilities, the first being to actually use it in a real adaptable SBA development project, the second being to compare it to a similar model that has been validated in a real scenario. The latter option was chosen mainly for practical reasons. While it is arguably better to validate the model in real scenario, there is also a lot to be learned by comparison with an existing approach. In this paper the comparison shows that the adaptation process model does contain the activities required to safely adapt an SBA during runtime.

The final threat considered is whether or not the data collected reached a point of saturation where activities and tasks identified

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

Table A.19
Adaptation activities.

Source	Adaptation activity	Generic activity
Lane et al.	Monitor message sequences among services and its partners	Runtime monitoring
Lane et al.	Runtime service discovery	Adapt SBA
Lane et al.	Requirements and analysis stage: define KPIs and management policies	Define Adaptation Requirements
Lane et al.	Service specification: identify the service properties to specify	Define Adaptation Requirements
Lane et al.	Specifying service decision model	Define Adaptation Strategies
Lane et al.	Set warning thresholds and alerts for compliance failures	Define Monitoring Requirements
Lane et al.	Specify monitoring rules according to the adopted SeCSE monitoring language (SECMOL)	Define Monitoring Requirements
Lane et al.	Service deployment: deploy the monitoring rules and recovery policies within the monitoring system	Deploy Monitoring Mechanisms
Lane et al.	Gather QoS metrics on the basis of SLAs	Design Monitors
Lane et al.	Evaluate SLA QoS metrics	Design Monitors
Lane et al.	Readjust service weights for request queues	Design Monitors
Lane et al.	Designing service adapters	Design Adaptation Mechanisms
Lane et al.	Recovery management: identify, by looking at the monitoring data, the needs for a recovery action	Event reasoning from Monitoring
Lane et al.	Detect protocol violations	Runtime monitoring
Lane et al.	Monitor service, application, middleware, OS, hardware, and network	Runtime monitoring
Lane et al.	Monitor workloads	Runtime monitoring
Lane et al.	Insert monitoring rules and recovery actions in concrete parts of the service composition executable description	Runtime monitoring
Lane et al.	Monitor services	Runtime monitoring
Bucchairone et al.	Define adaptation and monitoring requirements	Define Adaptation Requirements/ Define Monitoring Requirements
Bucchairone et al.	Design for monitoring and adaptation	Design Monitors/Design Adaptation Mechanisms
Bucchairone et al.	Construction of monitors and adaptation mechanisms	Implement Monitoring and Adaptation Mechanisms
Bucchairone et al.	Deployment-time adaptation	Define Adaptation Requirements
Bucchairone et al.	Run-time Monitoring	Runtime monitoring
Bucchairone et al.	Decide between adaptation and evolution	Define Adaptation Requirements
ProDAOSS	Organisational services realization paths are documented by a dynamic service hypergraph	Define Adaptation Requirements
ProDAOSS	Organisational services are designed as service centers in the architectural design discipline	Define Adaptation Requirements
ProDAOSS	Services realization environment is open and adaptable through the use of a reinforcement learning algorithm and a probabilistic reputation model	Event reasoning from Monitoring
PLASTIC	Runtime Analyser	Runtime monitoring
PLASTIC	SLA monitor	Runtime monitoring
PLASTIC	On-line validation	Runtime monitoring
PLASTIC	Evolution policies decoder	Event reasoning from Monitoring
BCDF	Take enterprises business and technical requirements as well as dependencies between them into consideration	Define Adaptation Requirements
BCDF	Enterprises describe their purpose and high level requirements for a business collaboration	Define Adaptation Requirements
BCDF	Define the operational conditions under which businesses can cooperate	Define Adaptation Requirements
BCDF	Negotiation agreement describing the interactions among the services from the different parties	Design Adaptation Mechanisms
Chang	Defining Target Services	Define Adaptation Requirements
Chang	Defining Unit Services	Define Adaptation Requirements
Chang	Planning service component acquisition	Design Adaptation Mechanisms
Chang	Acquiring service components	Design Adaptation Mechanisms
Chang	Composing services	Adapt SBA
Multi-view SOAD	Develop use case models	Define Adaptation Requirements
Multi-view SOAD	Service identification by viewpoint	Define Adaptation Requirements
Multi-view SOAD	Service interface conception	Design Adaptation Mechanisms
Multi-view SOAD	Mapping to platform specific models and code generator	Event reasoning from Monitoring
CSOMA	Model services with variability points	Define Adaptation Requirements
CSOMA	Select orchistration schema based on the context of the incoming request	Event reasoning from Monitoring
CSOA	Design adaptation views for Platform Specific Models (PSMs)	Define Adaptation Requirements
Dino	Develop UML2 Modes Model	Define Adaptation Requirements
Dino	Generate requirements and capabilities of broker services	Define Adaptation Requirements
Dino	Set dynamic service broker configuration	Design Adaptation Mechanisms

begun to re-occur in the later interviews. The later interviews did repeat much of the earlier findings with the exception of the life-cycle processes in the adaptation cycle. Had the interview questions relating to the adaptation cycle been focused on more during the interviews this would have resulted in more data. The problem seemed to be that the questions for the evolution cycle often took a lot of time and by the time the adaptation cycle questions came around the respondents enthusiasm diminished resulting in less data being collected. A solution may have been to focus solely on the adaptation cycle in later interviews, however, this would have been at the cost of valuable data collected by the evolution related questions. We plan to take this approach in future research.

8.1.1. Validation

Lincoln and Guba [36] argue that qualitative research should be internally valid, externally valid, reliable and objective. In this paper we focused on internal and external validity. Internal validity relates to "how" the research is carried out, and whether the methods used are credible. Two suggested methods of ensuring internal validity are data source and method triangulation [37]. Externally

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003

validity refers to the transferability of the research results to other similar contexts. If research results are transferable then they are arguably much more useful than results that are specific to a particular context.

The internal validity of this research has been sought through the use of data source and research method triangulation. There were two data sources used to construct the process model, SOAdapt-FoR was constructed with data from relevant literature sources, while the empirically grounded version of the model was constructed using data gathered during field work. There were also two research methods employed, survey research where individuals were interviewed and case study research where a the development process of SbaSoft was documented through interviews with several practitioners working at SbaSoft.

8.2. Conclusions and future work

During this research, the authors conducted a review of the literature and identified adaptation activities that could be used to adapt SBAs. These activities in combination with a skeleton life-cycle model proposed by the S-Cube consortium [18] formed the basis for a frame of reference process model for adapting SBAs. This frame of reference was used to guide interviews with SBA development practitioners who had experience with or who could provide expert opinion on how to adapt SBAs. The data that was collected in these interviews was transcribed and analyzed using qualitative content analysis techniques. The resulting adaptation activities and tasks were constructed into a detailed process model identifying the relevant stakeholders and development artifacts for each stage of the process. The model's transferability and capability were demonstrated during an evaluation process where the model was systematically compared to a component-based application adaptation model and an empirically based SBA development life-cycle. The approach taken in this paper has advantages over similar approaches in that its process focused and is based on input provided by experts from the field.

Future work will focus in refining the process model and investigating whether or not the activities identified for the adaptation cycle can be enriched with more detail. In order to determine the applicability of the model in the field, efforts will be made to use the model during an adaptable SBA development project.

Appendix A. Adaptation activities

See Table A.19.

References

[1] Rational, Rational unified process – best practices for software development teams, Tech. Rep. TP026B, 1998.
[2] M.P. Papazoglou, W.v.d. Heuvel, Service-oriented design and development methodology, Int. J. Web Eng. Technol. 2 (4) (2006) 412–442. URL <http://portal.acm.org/citation.cfm?id=1358575.1358582>.
[3] A. Arsanjani, S. Ghosh, A. Allam, T. Abdollah, S. Gariapathy, K. Holley, SOMA: a method for developing service-oriented solutions, IBM Syst. J. 47 (3) (2008) 377–396. URL <http://portal.acm.org/citation.cfm?id=1466610.1466613>.
[4] K. Mittal, Service oriented unified process, 2009, <http://www.kunalmittal.com/html/soup.html>. URL <http://www.kunalmittal.com/html/soup.html>.
[5] M. Trainotti, M. Pistore, G. Calabrese, G. Zacco, G. Lucchese, F. Barbon, P. Bertoli, P. Traverso, Astro: supporting composition and execution of web services, in: Lecture Notes in Computer Science, vol. 3826, 2005, p. 495.
[6] D. Linner, H. Pfeffer, I. Radusch, S. Steglich, Biology as inspiration towards a novel service life-cycle, in: International Conference on Autonomic and Trusted Computing (ATC 2007), 2007, p. 94102.
[7] S. Lane, I. Richardson, Process models for service based applications: a systematic literature review, Information and Software Technology (2011).
[8] Y. Wautelet, Y. Achbany, J. Lange, M. Kolp, A process for developing adaptable and open service systems: application in supply chain management, in: International Conference on Enterprise Information Systems (ICEIS 2009), Lecture Notes in Business Information Processing, vol. 24, Springer, Milan, Italy, 2009, pp. 564–576.
[9] L. Oconner, B. Orriens, J. Yang, M. Papazoglou, A rule driven approach for developing adaptive service oriented business collaboration, in: International Conference on Service-Oriented Computing (ICSOC 2005), Lecture Notes in Computer Science, vol. 3826, 2005, pp. 182–189.
[10] S. Vale, S. Hammoudi, Model driven development of context-aware service oriented architecture, in: The 11th IEEE International Conference on Computational Science and Engineering – Workshops, 2008, pp. 412–418.
[11] T. Margaria, B. Steffen, M. Wirsing, M. Hoelzl, L. Acciai, F. Banti, A. Clark, A. Fantechi, S. Gilmore, S. Gnesi, L. Goenczy, N. Koch, A. Lapadula, P. Mayer, F. Mazzanti, R. Pugliese, A. Schroeder, F. Tiezzi, M. Tribastone, D. Varro, SENSORIA patterns: augmenting service engineering with formal analysis, transformation and dynamicity, in: Third International Symposium on Leveraging Applications of Formal Methods, Verification and Validation, Communications in Computer and Information Science, vol. 17, 2008, pp. 170–190.
[12] G. Spanoudakis, A. Zisman, A. Kozlenkov, A service discovery framework for service centric systems, in: IEEE International Conference on Services Computing, 2005, vol. 1, 2005, pp. 251–259.
[13] K. Verma, K. Gomadam, A.P. Sheth, J.A. Miller, Z. Wu, The METEOR-S approach for configuring and executing dynamic web processes, LSDIS METEOR-S Project Technical Report, 2005.
[14] A. Bucchiarone, C. Cappiello, E. di Nitto, R. Kazhamiakin, V. Mazza, M. Pistore, Design for adaptation of service-based applications: main issues and requirements, Stockholm, Sweden, 2009.
[15] P. McKinley, S. Sadjadi, E. Kasten, B. Cheng, Composing adaptive software, Computer 37 (7) (2004) 56–64, doi:10.1109/MC.2004.48.
[16] S. Consortium, S-Cube knowledge model, 2011, <http://www.s-cube-network.eu/km>. URL <http://www.s-cube-network.eu/km>.
[17] J.C. Derniame, B.A. Kaba, D. Wastell, Software Process: Principles, Methodology, and Technology, Springer Verlag, 1999.
[18] S. Consortium, State of the art report on software engineering design knowledge and survey of HCI and contextual knowledge, Tech. Rep. PO-JRA-1.1.1, July 2008.
[19] P. Herzum, O. Sims, Business Components Factory: A Comprehensive Overview of Component-Based Development for the Enterprise, John Wiley & Sons, Inc., 2000.
[20] R. Haesen, M. Snoeck, W. Lemahieu, S. Poelmans, On the definition of service granularity and its architectural impact, in: Z. Bellahsne, M. Lonard (Eds.), Advanced Information Systems Engineering, vol. 5074, Springer, Berlin Heidelberg, 2008, pp. 375–389. URL <http://www.springerlink.com/content/g17h73u685227g63/>.
[21] P. Oreizy, M. Gorlick, R. Taylor, D. Heimhigner, G. Johnson, N. Medvidovic, A. Quilici, D. Rosenblum, A. Wolf, An architecture-based approach to self-adaptive software, IEEE Intell. Syst. Appl. 14 (3) (1999) 54–62, doi:10.1109/5254.769885.
[22] F. Ahlemann, H. Gastl, Process model for an empiracally grounded reference model construction, in: P. Fettke, P. Loos (Eds.), Reference Modeling for Business Systems Analysis, Idea Group Publishing, 2007.
[23] S. Lane, Q. Gu, P. Lago, I. Richardson, Adaptation of service based applications: a maintenance process?, Tech. Rep. Lero-TR-2010-08, Lero, the Irish Software Engineering Research Centre, Limerick, Ireland, 2010. URL <http://www.lero.ie/sites/default/files/files/Lero-TR-2010-08.pdf>.
[24] A. Bucchiarone, R. Kazhamiakin, C. Cappiello, E. di Nitto, V. Mazza, A context-driven adaptation process for service-based applications, in: Proceedings of the 2nd International Workshop on Principles of Engineering Service-Oriented Systems, 2010, pp. 50–56.
[25] IEEE, IEC, ISO, ISO/IEC 15288, systems and software engineering – system life cycle processes, Tech. Rep., 2008.
[26] C. Robson, Real World Research: A Resource for Social Scientists and Practitioner-Researchers, Wiley-Blackwell, 2002.
[27] R.K. Yin, Case Study Research: Design and Methods, Applied Social Research Methods Series, third ed., vol. 5, Sage Publications, Inc., 2002.
[28] M.B. Miles, A.M. Huberman, Qualitative Data Analysis: An Expanded Sourcebook, SAGE publications, Inc, 1994.
[29] I. Griffin, I. Richardson, Using LATEX for qualitative data analysis, The PracTEX Journal, 2010-1.
[30] P. Oreizy, N. Medvidovic, R. Taylor, Architecture-based runtime software evolution, in: Proceedings of the 1998 International Conference on Software Engineering, 1998, pp. 177–186. doi:10.1109/ICSE.1998.671114.
[31] S. Durvasula, M. Guttmann, A. Kumar, J. Lamb, T. Mitchel, B. Oral, Y. Pai, T. Sedlack, H. Sharma, S. Sundaresan, Introduction to Service Lifecycle, SOA Practitioners Guide. Part 3, 2007.
[32] Business process modeling notation (BPMN) information, 2010, <http://www.bpmn.org/Documents/FAQ.htm>. URL <http://www.bpmn.org/Documents/FAQ.htm>.
[33] P. Harmon, Second generation business process methodologies, Business Process Trends 1 (5) (2003).
[34] W.W. Royce, Managing the development of large software systems, in: Proceedings of IEEE Wescon, vol. 26, 1970, p. 9.
[35] B.G. Glaser, Basics of Grounded Theory Analysis, Sociology Press Mill Valley, CA, 1992.
[36] Y.S. Lincoln, E.G. Guba, Naturalistic Inquiry, Sage Publications, Inc., 1985.
[37] P. Liamputtong, D. Ezzy, Qualitative Research Methods, second ed., Oxford University Press, USA, 2005.

Please cite this article in press as: S. Lane et al., SOAdapt: A process reference model for developing adaptable service-based applications, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.10.003