Applying Component Concepts to Service Oriented Design: A Case Study

Balbir Barn and Samia Oussena

Thames Valley University, Computing Department
Wellington Street, Slough, SL1 1YG, UK

Abstract. This paper argues that appropriate modeling methods for service oriented development have not matured at the same pace as the technology because the conceptual underpinning that binds methods and technology has not been sufficiently articulated and developed. The paper describes an adaptation and enhancement of component based techniques to support the development of a service oriented method. As a result of the evaluation of using component concepts to support service oriented design, an integrated conceptual model describing how concepts from services and components are related is presented. The experimental data derives from a complex case study from the Higher Education Enterprise arena.

1 Background

Currently there is a focus on enterprise application integration using distributed architecture principles and in particular, there is a convergence to so-called Service Oriented Architecture (SOA) for application design and integration [19]. While enterprise systems have attained a degree of technical integration in many cases, the full benefits of business integration that could be gained from seamless support of business processes may only be partially realized. The developing principles around SOA are placing importance on a solid understanding of business processes and aligning developed or procured services to support those processes [11]. Thus methodologies that can support process led application development and assembly will acquire greater utility.

1.1 The Problem

Service Oriented Architecture is a disruptive technology because of the opportunity it provides to rethink the way systems are created and evolved. However there is still a relative lack of robustly applied and practical methodology support for such an approach. This observation has also been noted by Quartel et al [22] where they observe that technological developments should be supported by modeling methods and languages to support service-oriented design. One example where the methodology issues have been addressed to some extent is the recent work by Erl where there has been an effort to recognize that service-oriented analysis is an important element in the design of effective SOA [9]. However, here, the focus has been to derive services from a business process orchestration specification.

In contrast, Component Based Development (CBD) has reached a level of maturity where there is a significant body of knowledge addressing methodology requirements as well as technological issues. Given this, the research question addressed in this paper is:

"Can component based development concepts, methods and techniques support service oriented design?"

A service based architecture presents multiple concerns or architectural viewpoints. Consequently, the focus of the research question is further refined to address the functional viewpoint – that is, what an application (based on SOA) must do in order to support the business requirements of the user. Thus example areas that are not addressed in this paper are: the issues that arise at the boundaries between design and deployment of service-centric systems; and binding of services based on run-time monitoring of service-centric systems.

1.2 The Contribution of this Paper

This paper outlines an approach to service oriented design by drawing on the lessons learnt and the best practices from component based practices. The focus of this paper is on service identification and partitioning of applications into services and how such techniques can be captured in model based form. Further, the paper proposes an approach to business process partitioning that provides a model based migration strategy to process implementation using technologies such as Business Process Execution Language (BPEL). The paper also contributes an integrative view of services, components and business processes to further emphasize the benefits of pursuing this particular approach. Some of the observations and evaluation of software tools applied in the methods outlined also indicate that there are issues of tool usage which can help to inform tool selection and deployment.

The remainder of this paper is structured as follows: The reader is introduced to the background case study informing this research in section 2. Section 3 addresses the core of the paper and focuses on the comparison of concepts underpinning both component modeling and service oriented architecture with reference to relevant literature. Section 4 provides an evaluation of some of the results in the context of the case study and provides further details on the integrated conceptual model. Section 5 concludes the paper and outlines areas for further work.

2 Case Study

This section provides a short description of the context of the case study for which the business process modeling and subsequent application design was performed.

The e-Framework [16] is an initiative by the U.K's Joint Information Services Committee (JISC) and Australia's Department of Education, Science and Training (DEST) to build a common approach to Service Oriented Architectures for education and research. As part of this initiative, in 2005, JISC requested projects to develop reference models for a number of domain areas. The work described in this paper is derived from one of the projects.

The Course Validation (CV) process is one of the most important business processes within Higher Education Institutions (HEIs) and between HEIs and other institutions. New courses and the continuation of existing courses are the direct outputs of this process.

Further, the process is case-based, knowledge centric and highly collaborative. Each instance of the process is a case and will focus typically on different subject domains and therefore require different knowledge bases and experts to support the process. Only the essential framework (the rules and governance) of the process remain standardized.

The scope of the application domain is as follows. Course Validation can include the specification of new courses at various levels (e.g. undergraduate and postgraduate). Course Specifications address areas such as rationale, appropriateness, justification, marketing analysis, resources required, economic viability of the courses, and detailed descriptions of the courses in terms of outcomes, aims and objectives and so on. Much of the scope of course validation is determined by local institutional constraints (e.g. relationship to other courses and university regulations) but there are wider requirements that impose a significant overhead on the developmental process for validating new courses. These wider requirements are determined by the UK national bodies such as the Quality Assurance Agency (QAA) [21].

Even though HEIs may differ in the implementation of business processes to support course validation the constraints imposed by external bodies such as the QAA provide some standardization for the validation process and its outputs. These constraints are a basis for defining a canonical business process for supporting course validation.

A case study approach to the problem was adopted as there are several examples in IS research where there is evidence that case study based methodologies are well suited for exploring business processes in an organizational setting. Examples include those described in Huang et al [12] and Sedora et al [23]. Case studies provide an opportunity to take an interpretivist stance on how the systems and structures in place are based on the meanings of concepts and how people use those concepts. A case study also allows in-depth exploration of issues. However, given the nature of the course validation process it was important to get an understanding of how different types of institutions implemented their own course validation processes. Consequently we explored in depth, the course validation processes at four institutions.

After a period of business analysis, process models of course validation processes at each of the institutions were constructed. Accompanying information and data supporting these processes were also modeled. All modeling at this stage was performed using the IBM Rational XDE Toolset. The visual models were evaluated and an approach to synthesizing the models from each institution into a single canonical model was developed and then applied. This approach includes rules for identifying variances between processes and is described in more detail elsewhere [3].

The result was a pair of canonical models for the process and the information which were used as input to the software design and implementation stages to develop a set of software services that allowed us to automate part of the business process.

The remainder of the paper focuses on how the canonical process and information models were partitioned into a set of services to support the software design phase using a component based design approach.

3 Related Work and Approach

The approach taken in this research is based on three key principles: Firstly, the importance of systemizing the relationship between Component Based Development (CBD) and Services; secondly, to consider application partitioning from the perspectives of both business process partitioning and data partitioning; and finally the requirement to articulate a unifying conceptual model that addresses methods, business processes, components and services. Also critical to the approach taken in this paper, was the need to ensure that a model driven approach was followed. This was accomplished by, ensuring as far as possible, that all activities, artifacts and transformations were performed within one or more software tools. As the evaluation section indicates issues emerged during this process.

The central thesis of the paper argues that Component Based Development (CBD) provides a natural evolution to service oriented architectures because of the conceptual similarities and overlaps between the two software architecture approaches. In this section, the conceptual mappings between components and services are presented based on review of existing work. These mappings indicate the strong correlation between these two approaches thus indicating that it is instructive to look to CBD for appropriate methods and techniques to apply to SOA.

One important strategic distinction between the two approaches is the focus of integration strategies to address heterogeneous application architectures. While CBD at least, conceptually can be used to provide an application architecture that makes it possible to mix different implementation technologies, the evidence to date has indicated that this capability has not really been taken advantage of. For example, there are software component libraries for the Microsoft platform and similar libraries for the J2EE platform. SOA, on the other hand, has at its heart, standards and technologies that support interoperability. The use of XML based standards and protocols such as XML, SOAP, WSDL and UDDI allows services to be implemented in a particular technology while allowing access to the service from varying technical platforms using the so-called "wire" standards. A core common concept underpinning both CBD and service oriented design is the notion of an interface specification – a precise description of the behaviour of a software implementation. Interfaces can be used to provide wrappers to existing applications / modules such that it is possible to continue to use legacy applications in new technological environments. SOA is particularly suited to this approach and is potentially the most significant benefit arising from adopting a SOA strategy by an organization.

However, while a component can conceptually support more than one interface, a service has only one interface (in WSDL 1.1). Additionally, WSDL does not provide a mechanism for representing detailed behavioural semantics such as pre/post condition pairs. In separate work Estier et al demonstrate similar mappings and observations and in fact also use similar terminology such as "core". Their focus was on providing a "Contract" basis to service design rather than model based specification and generation [8]. Other work has developed a UML profile for describing WSDL (and therefore Web Services) to support WSDL generation from UML models – although the reported work proposes that implementation of add-ins for WSDL generation to commercial products such as Rational Rose is part of planned work [18]. Nonetheless this work while mapping to UML and not components would appear to further substantiate the validity of looking to CBD practice for methodology support.

The mappings indicate that we can leverage approaches and maturity of CBD practice to the design and implementation of web services by tailoring existing methods for software development.

Probably the most refined and detailed articulation of a component based method is work done by D'Souza and Wills in their description of Catalysis [7]. This method was later used to underpin the development of Cool:Spex [1, 2] (an early product to focus explicitly on application design using component based principles) and also other CBD methods [5]. Some of the CBD techniques from Catalysis and their derivatives that we can use include: component identification (or application partitioning), component specification, and component dependency management.

Application partitioning – the act of identifying discrete pieces of functionality into independent chunks of software is core to notions of component based development. However, CBD assumes a data centric view of partitioning. For example Erl describes such components as Entity Services. One proven approach to component partitioning and therefore service partitioning is that described by Cheesman and Daniels [5]. Their approach is based on using the information model (business concept model) as a starting position from which to make application partitioning decisions using an algorithm based on identifying core types and other types related to the core type.

However, despite the detail, complexity and scope available in the Catalysis method (and its variants), there has been relatively light attention to process modeling. This lesser emphasis is critical as process modeling is a crucial element for SOA where the orchestration of services to support an application is central to application assembly. The substantial standardization effort in business process execution (BPEL4WS) and prevalence of tools for choreographing applications from a set of services provides an indication of its importance.

This paper proposes that while data centric partitioning is one concern, it is also necessary to have a parallel and equivalent view of process partitioning. When processes are long, complex and require significant human intervention at various stages then the need for process decomposition is even more transparent. The case study used in this paper illustrates this point.

The Rational Unified Process (RUP) [14, 15] provides some guidance to this vis a vis the distinction business use cases standard use cases. This is an example of process decomposition. However, there is in-sufficient guidance and somewhat ambiguous rules for how business use cases map to use cases. Other ways of managing the modeling of process complexity for example to use roles – i.e. focusing only on the activities and their collaborations performed by specific roles [20] were considered inappropriate because the underlying process model was based on transformation (that is: input transformed by activity to output) rather than more communication/coordination views of processes.

One established approach is the use of "Event Consequences" – that is a business event is a trigger to a sequence of activities that are performed in response to the event. There is a rich body of knowledge which supports the notion of business process understanding using this approach for example [6, 24]. The set of activities that are triggered can then be viewed as a sub-process of the overall business process. Such a sub-process provides a better level of granularity for describing analysis scenarios for support the design and implementation stages of a software development process. An additional benefit of using events to partition a business process is the potential direct modeling transformation into BPEL specifications where there are modeling concepts for supporting events and their subsequent triggering of consequences of actions.

Summarizing, CBD practice provides a useful conceptual toolbox that need to be enhanced with a more detailed treatment of process modeling techniques in order to be useful to application development using SOA.

4 Case Study Example and Evaluation

Much of the anticipated benefits of a SOA approach are assumed to be the rapid assembly via orchestration of applications comprising one or more services. A pre-requisite for orchestration is a detailed understanding of the business process to be supported and the (ideally) model based specification. Thus the starting premise of the project approach is to precisely describe the business process using an appropriate modeling toolset.

The business analysis phase for the Course Validation (CV) domain produced two complex models – the process model and the information model.

4.1 Process Partitioning

Given the CV process complexity in particular, a way for decomposing the process into more manageable sub-processes was required. The CV process already had natural groupings of activities (these were distinct stages in the process) however, even these groupings were difficult to manage and it was necessary to define smaller sub-processes.

When a business process is a type of collaborative case process such as Course Validation then an especially useful form of partitioning is to identify situations in the business process where there is delay in the process because there is a need for an external event to occur. Once the event has arisen, new activities are undertaken. These groupings of activities are treated as sub-processes.

To support these sub-processes user scenarios were also developed. A user scenario is an evocative way of instantiating a route through a part of the business process. User scenarios are effective in extracting requirements because they express functionality in the language of users [4, 25] and for the implementation phase in this project they also provided additional context to development staff who were not involved in the original analysis stages of the project. During the development phase – the implementation team found the scenarios useful but still needed to elaborate additional sequence diagrams to further identify operation requirements.

Thus, in our process modeling we introduced the notion of sub-process scenarios. A sub-process scenario comprising one or more activities is triggered by an event such as a time or data event. This scenario and its accompanying user story can then be analyzed by the software designer to identify operations and allocate them to specific components/services.

4.2 Service Identification

Identification and modeling of services is the core of what is presented in this paper. It is argued that there is lack of methods and techniques to support service identification and modeling and currently most effort is focused advice and guidance on programming issues. It would appear that modeling advice is largely derived from object oriented analysis. Here it is proposed that given the conceptual closeness between services and components, it is possible to utilize techniques from CBD. In essence, the domain model or information model in our example is partitioned into "components" by firstly identifying types which are deemed to be core – that is business types or objects which are essential to the organization and then traversing associations to other types that are detailing – that are providing additional details to the core type. This subsetting provides a natural component boundary. Each component identified is then allocated an interface type which will house the operations for the component. This model thus corresponds to the service as follows: Component Interface is equivalent to Service; Core Type and detailing type are equivalent to the Port Types with their associated Messages and subsequently the elements and their schema. This approach bears comparison with Estier et al. who have similarly used CBD principles in their work on service contracts.

During the service partitioning activity a number of rules / hints emerged – the use of which has the potential for better quality component / service models. For example if two core types are related by a mandatory association (at both ends) it is still better to treat the core types as housed in separate components.

Often, components have a cross-relational dependency manifested via an intersecting detailing type. When this pattern occurs, one relationship is often "specifying" and the other is a "usage". In such situations, the intersecting type should always become the detailing type in the component that owns the "specifying" relationship. We anticipate further useful rules and hints as we continue to refine the component models.

4.3 Process Led Application Assembly

Once the components / services have been identified in this manner, the sub-process scenarios and their accompanying textual narratives are used to map activities from the process to an operation on a service. Service responsibility is based primarily on determining the types (information) being manipulated in the process and then allocating the service behaviour to the component/service that owns that type. The results of applying this technique to each of the sub-process scenarios is a set of components/services with behaviour allocated to them. In theory, then each component/service model can then be used to generate the required WSDL specification to support the web service.

In practice, this is where the violation of core model driven development principles unfolded. As intimated earlier, our key goal was to take a model driven approach to specify a business process and the accompanying information model; partition the models into a set of services; and then assemble services together to implement the business process. In order to do this, we would use software tools to model and generate the required elements. During the specification and design work toolset issues meant that the domain modeling (business analysis) was done using IBM Rational XDE as it was clear that the preferred toolset IBM Rational Software Architect (RSA) was not yet mature enough with its support for UML 2.0. However, the RSA implementation environment was considered to be superior to XDE so when the business analysis modeling and the partitioning into components/services was completed, the XDE models were imported without loss of data into RSA. Further modeling refinements were undertaken within RSA.

Issues during the design modeling phase also made it clear that it would have been better to create separate RSA models of each component (within the same overall project) as it made generation of WSDL and other XML easier and less error prone.

On generation to WSDL services, it became apparent that while the model structure to represent a component/service was correct (in that, all the required information to generate a WSDL spec was present), WSDL generation was not possible and the only generation of data that was achieved was the XSD schemas for the data requirements of the services. This represented a significant drawback to our proposed approach and the team is currently investigating alternative methods of WSDL generation with RSA..

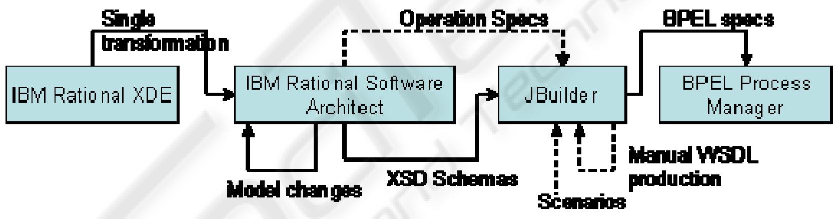

Fig. 1. Toolsets and transformations of models.

JBuilder was selected as the preferred toolset for designing the BPEL processes as again RSA and the Eclipse Plugin for BPEL process design did not fully support the design requirements. In this case the user interactions (user tasks) were not supported by the Eclispe Plugin. Within JBuilder, WSDL was handcrafted using further data from the text based user stories.

5 An Integrated Model for Component and Service Method Concepts

Having described the methodology for identifying and describing services from a business process basis, this section now proposes how the methodology and concepts need to be integrated to produce an overarching conceptual model which can be used to provide a basis for method refinement, tool construction and good practice.

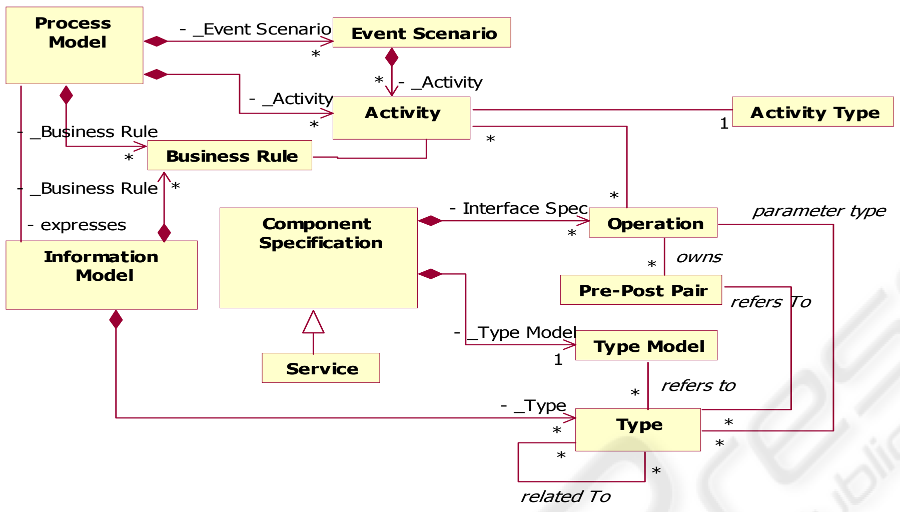

Fig. 2. SOA Integrated model.

The diagram above provides a UML model of the principal concepts involved. The Process model is decomposed into a set of Event Scenarios which are themselves a grouping of activities which have an ordering defined ultimately by UML semantics for activity modeling. An Event Scenario or Sub-Process provides a natural mapping to BPEL workflows.

In parallel, the domain information model is partitioned into a set of Services using CBD practice. A service has a set of operations which may or may not be specified by pre/post specification pairs. The types used by the operations of a service are grouped by the notion of an interface type model. These types can be used as the XSD schema for a WSDL specification. Activities are mapped to operations on the services via the Service/Activity matrix (or sequence diagram). Activities are classified as either manual (therefore not implemented, but their interfaces are specified), event receipt or normal. This classification is modeled by the Activity Type concept. This paper has not discussed business rules (constraints or invariants), in detail. In general, during the analysis phase, business rules were captured using Constraint annotations in the various models. However, we are planning further work to more formalize the capture of business rules using technologies such as XRules.

A further benefit of using an underlying meta model to describe methods is that development of techniques and concepts can be engineered allowing tailoring of method elements to support specific project requirements [13].

6 Conclusions and Further Work

This project set out to explore the interaction between SOA and model driven development. The case study and modeling approach has demonstrated that there is sufficient conceptual equivalence between component based approaches and service oriented architecture to warrant the use of CBD methods to identify and model services. From a complex process model, it was possible to partition the process into manageable sub-processes which could be orchestrated as BPEL workflows (albeit handcrafted).

The selection and deployment of the particular set of tools used in the project have been used to implement services and their process definitions with some success – with the primary problem centred around the model based generation to WSDL specs. The overheads and risks incurred by the use of such tools for bespoke application development using SOA remain significant and it is not clear how successful such a tool deployment strategy would be. It is possible that further investigation and increased expertise in tools such as RSA could help mitigate these risks.

Consequently, the project team is minded to conclude that SOA remains a significant challenge and perhaps best suited to application integration rather than bespoke development. As a result of this experiment, further work is being planned on the use of Business Process Management Toolsets such as Intalio Designer. One potential use of the conceptual model (after further research and validation) presented in figure 1 could be its use as a evaluation tool for the selection of tools (single or combined) However, SOA does require an emphasis on a business process modeling and research presented in this paper provides some enhancements to process modeling to ease the move from CBD to SOA. As we continue to develop services from new sub-process scenarios it is likely that we will refine our component partitioning strategy and the rules and hints to support the strategy. The use of the sub-process scenarios as model based input to Business process execution (BPEL) will also be the subject of further evaluation and study.

References

1. Barn, B.S., Brown, A.W., Cheesman, J.: Methods and Tools for Component Based Development. In Tools 98: Technology of Object-Oriented Languages and Systems, (1998)
2. Barn B.S., Brown A.W. Enterprise-Scale CBD: Building Complex Computer Systems from Components. In: 9th International Conference on Software Technology and Engineering Practice (STEP'99), Pittsburgh, Pennsylvania, USA (1999)
3. Barn, B.S., Dexter, H., Oussena, S., Petch, J. An Approach to Creating Reference Models for SOA from Multiple Processes In: IADIS Conference on Applied Computing, Spain (2006)
4. John Carroll. “Five Reasons for Scenario-Based Design” in Proceedings of the 32nd Hawaii International Conference on System Sciences – 1999.
5. Cheesman, J., Daniels, J. UML Components. Addison-Wesley (2001)
6. Cook, S., Daniels, J. Designing Object Systems: Object-oriented Modelling with Syntropy. Prentice Hall (1994)
7. D'Souza, D. F., Wills, A. C. Objects, Components, and Frameworks with UML: The Catalysis Approach. Object Technology Series. Addison Wesley, Reading Mass., (1999)
8. Estier, T., Michel, B., Reinhard, O. Modeling Services using Contracts: Identifying dependencies in Service-Oriented Architectures. In: EMMSAD 2006 Workshop – CAISE (2006).

9. Erl, T. Service Oriented Architecture – Concepts, Technology and Design. Prentice-Hall, USA (2005).
10. Frankel, D. Model Driven Architecture, OMG Press (2004)
11. Frankel, D.: Business Process Trends. BPTrends http://www.bptrends.com/ publicationfiles/07%2D05%20COL%20BP%20Platform%20%2D%20Frankel%2Epdf (2005)
12. Huang J.C., Newell S., Poulson B., Galliers R.D. Deriving Value from a Commodity Process: a Case Study of the Strategic Planning and Management of a Call Center. In: Proceedings of the Thirteenth European Conference on Information Systems (Bartmann D, Rajola F, Kallinikos J, Avison D, Winter R, Ein-Dor P, Becker J, Bodendorf F, Weinhardt C eds.), Regensburg, Germany. (2005)
13. Henderson-Sellers, B. Method engineering for OO systems development. Comm.. ACM 46, 10 (Oct. 2003), 73-78. DOI= http://doi.acm.org/10.1145/944217.94424
14. IBM-Rational: The Rational Unified Process (RUP),http://www-306.ibm.com/software/awdtools/rup/ (2001)
15. Kruchten, P. Rational Unified Process, Addison Wesley (1999)
16. E-learning Framework: http://www.elframework.org/ (2006)
17. Low, G. C., Rasmussen, G., Henderson-Sellers, B. Incorporation of distributed computing concerns into object-oriented methodologies; Journal of Object-Oriented Programming. (1996) Vol. 9, no. 3, pp. 12-20
18. Esperanza Marcos, Valeria de Castro, and Belén Vela (2003) "Representing Web Services with UML: A Case Study". In M.E. Orlowska et al. (Eds).: IC-SOC 2003, LNCS 2910, pp.17-27, 2003.
19. Ort, E. "Service-Oriented Architecture and Web Services: Concepts, Technologies, and Tools" http://java.sun.com/developer/technicalArticles/WebServices/soa2/ (2005)
20. Ould, M. A.: Business Process Management: A Rigorous Approach, BCS, ISBN: 1-902505-60-3 (2005)
21. QAA: http://www.qaa.ac.uk/
22. Dick Quartel, Remco Dijkman and Martin van Sinderen. "Methodology Support for Service-oriented Design with ISDL", ICSOC, 2004.
23. Sedera W., Rosemann M., Doebeli G. A process modelling success model: insights from a case study. In Proceedings of the Eleventh European Conference on Information Systems (Ciborra CU, Mercurio R, de Marco M, Martinez M, Carignani A eds.), Naples, Italy. (2003)
24. Texas Instruments. A guide to Information Engineering using the IEF™. TI Part Number: 2739756-0001. (1990)
25. Van Helvert, J & Fowler, C.J.H. (2004) Scenario-based User Needs Analysis. In Ian Alexander and Neil Maiden (eds) Scenarios & Use Cases Stories through the System Life Cycle. Wiley: London