50

AN E-SERVICE SOA MODEL FOR VIRTUAL SERVICE ENTERPRISES

Christian Zirpins and Wolfgang Emmerich
Computer Science Department, University College London, UK
(C.Zirpins|W.Emmerich@cs.ucl.ac.uk)

Economic theory defines services as customisable, interactive processes that providers have the potential to carry out together with clients that benefit from their effects. It is understood that service transactions are best organised by means of virtual collaborative networks, where ICT allows configuring multiple providers and processes on a per-request basis. Existing conceptual models for virtual service enterprises propose service virtualisation to allow for flexible and agile regulation and enforcement of coordination between multiple providers and clients. In this paper, we present an approach for realising business service virtualisation based on software service technology. In particular, we propose a SOA model for representing virtual business service processes as e-services. E-service models specify interactions between multiple providers and clients of virtual service enterprises by means of patterns and allow for flexible regulation and enforcement of their coordination.

1. INTRODUCTION

Services make up a growing industry of intangible goods. It has been argued that organisational requirements to provide configurable, interactive and immaterial service processes are best met by virtual organisation [Picot and Neuburger 1998]. Respective virtual service enterprises are temporal organisational networks that abandon institutionalised network management in favour of information and communication technology (ICT). This allows for dynamic identification, initiation, negotiation operation and liquidation of service networks on a per-request basis.

As an important aspect of ICT support for virtual service enterprises, regulation and enforcement of coordination rules for cooperative activities of service client(s) and providers in temporal service networks need to be enabled in a flexible and agile manner. In earlier work, we have shown, how this can be archived by virtualisation of services themselves [Zirpins and Emmerich 2008b]. We have presented a conceptual reference model that explains how to coordinate a virtual production network (VPN) for services by means of planning and control production of virtualised business services (VBS). The latter procedures can be effectively supported by ICT. We showed that respective technology includes representations of virtual service processes (e-services) as well as structured methods for their planning and control (e-service management). Moreover, we presented evidence for the fitness of software service technologies [Papazoglou and Georgakopoulos 2003] and Web Services [Alonso et al. 2004] in this respect. Software service abstractions showed potential to represent virtual business service processes in a way that service oriented development life-cycle methodology could be adopted to regulate and enforce coordinative rules in virtual business service production networks (VBSPN).

In the PARIS research project (**P**attern-based **AR**chitectures for **S**ervice **I**nteraction – see http://www.cs.ucl.ac.uk/staff/c.zirpins/paris/), we have developed an approach to realise e-service technology by means of service-oriented software architecture (SOA) for e-services and service-oriented development life-cycle methodology for e-service management. In terms of e-services, we propose a general SOA model that defines how virtual business service processes (VBSP) according to our conceptual model are to be represented by software service abstractions. More explicit, we provide a formal e-service metamodel in UML that defines concepts of workflow, software service interaction processes as well as business service interaction underlying our e-service abstraction. The metamodel underlies the definition of a domain specific graphical modelling language for e-services that can be used to design and execute e-services in the course of planning and control of VBSPs.

In this paper we focus on the PARIS e-service metamodel and its SOA for representing VBSPs. Section 2 introduces the software architecture model that has been developed to realise e-services. This includes an informal systems model as well as a formal metamodel that defines structure and semantics of e-service design in UML. The paper closes with related work and a summary and outlook in sections 3 and 4.

2. SERVICE-ORIENTED ARCHITECTURE FOR E-SERVICES

Service virtualisation technology builds on realisation of virtual service processes. These VBSPs essentially act as interaction programs to flexibly regulate and rapidly enforce coordination of cooperative activities between members of virtual service networks. To archive this, VBSPs substitute interactive aspects of conventional service processes by ICT. The goal is for representations of communication endpoints and interaction process patterns to allow for formal specification and automated enforcement of interaction and coordination between operative information systems of network clients, providers and brokers. This ability shall then be used to integrate systems of network participants into temporal cooperative information systems for control of service production that we refer to as *e-service systems*. Generally, we refer to the integrative core of such a system as *e-service*. Furthermore, we distinguish regulation of composition logic by means of *e-service schemas* from enforcement of composition logic by means of *e-service instances*.

Our general concept is to leverage service-oriented software technology to realise e-service systems in general and e-services in particular. More precisely, we adopt SOA for e-services and service-oriented development life-cycle methodology for e-service management. In this paper, we focus on SOA for e-service instances and on design of e-service schemas as SOA models. In the following, we will first outline a specific *service-oriented software model* that explains the roles, functions and software service abstractions involved in the life-cycle of e-service systems. We will then present an UML-based *e-service metamodel* that gives a detailed description of SOA for e-service instances and builds the foundation of a graphical language for e-service schemas.

2.1 A Basic Model for Service-Oriented E-Service Systems

Service-oriented computing (SOC) is a paradigm for building and operating software application systems by means of service abstractions and architecture, utilising service development life-cycle methods and methodology. The paradigm is fundamentally described by "service-oriented models" (SOM) like that of [Papazoglou and Georgakopoulos 2003]. SOMs specify roles (e.g. client, provider, broker, aggregator, operator) and their functions (e.g. description, publishing, discovery, selection, access, coordination, composition, marketing, support etc.) with respect to service development on various abstraction levels (basic, composed, managed). In our approach to realise VBS by means of SOC technologies, we propose a SOM refinement for the case of managing e-service-systems. Our SOM corresponds to the PARIS VBSPN model and we refer to it as *PARIS SOM (PSOM).* PSOM refines service abstractions, roles and role functions in order to match the requirements for service virtualisation in the PARIS conceptual model.

Firstly, PSOM refines software service abstractions for business service process virtualisation. The PARIS model defines *demands*, *assets* and *capabilities* as the main concepts of VBSPs. In PSOM, we map these concepts to refined software service abstractions. Demands and assets represent endpoints of clients and providers that handle interactions for consumption of service content. We map both concepts to refinements of atomic service abstractions referred to as *demand* and *asset service*. Demand and asset services are formally specified by service description languages and automatically controlled by means of service access mechanisms. We require specification and automation of service interaction steps by means of operational interfaces. Further specifications might include conversations or semantics.

Capabilities represent interaction process patterns of providers and brokers. They regulate and control an administrative procedure to access service content. We map this concept to a refinement of composite services referred to as *capability services*. Capability services are formally specified by means of service coordination protocols and automatically controlled by means of service composition schemas. We require specification and automation of service interaction processes by means of functional process structure, roles and implementation. Further specifications might include non-functional aspects. As regards concepts of flexible planning, we require specification of protocol patterns and their automated resolution into executable orchestration schemas. Such technologies are beyond current SOC state-of-the-art, but we propose a solution in our e-service metamodel. With respect to the structure of VBSPs, asset services are associated with local capability services that regulate and enforce interactions with demand services to access service contents. Furthermore, global capability services regulate and enforce interactions between local capability services to combine service content of various providers in a service network.

Fig.1 shows the refined service abstractions of PSOM and their relationships. Furthermore, it indicates the refinement of SOM roles and functions to match the roles of the PARIS conceptual model and their responsibilities of VBS production planning and control. Here, client- and provider roles of PSOM are both refinements of SOM service providers, as they provide atomic demand- and asset services. The coordinator role is a refinement of the SOM service aggregator, as it provides composite capability services. The abstract coordinator role is assigned to the concrete roles of broker and provider as regards global and local capability services.
In the pre-contact phase of VBS, PSOM roles are responsible for specification and mutually adjusting their services. Providers describe asset- and local capability ser-

vices and publish them in repositories of brokers that are accessible within and outside of the service network. Outside of the network, clients query the repository to discover asset services and capability services. In turn, they specify their demand services and likewise publish them in a repository. Inside the network, brokers discover demand services as well as asset services and related local capability services. They specify global capability services that combine asset services with respect to demand services by composing respective local capability services.

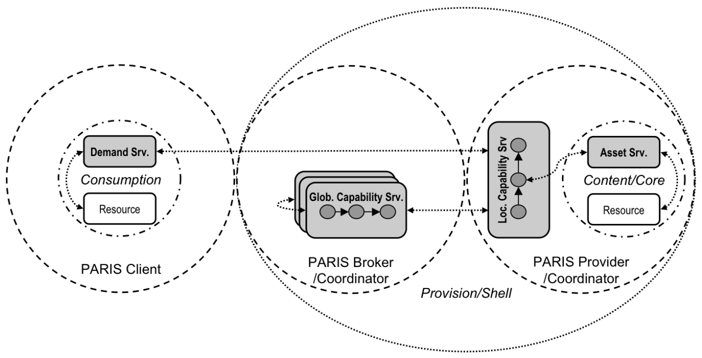

Figure 1 – Service-oriented Model of Business Service Virtualisation in PARIS

In the contact phase of VBS, roles of PSOM execute their services to automate control of business service processes. Brokers rapidly optimise and implement global capability services with respect to non-functional requirements of client requests. All other services can be implemented and deployed in the pre-contact phase. In turn, execution of global and local composite capability services automatically enforces the planned regulation of interactions in the business service process.

Finally, in the VBS post-contact phase, PSOM roles analyse service interaction logs that were monitored in the contact phase. With respect to the results, they might evolve specific aspects of the service specifications to optimise effectiveness or efficiency of the VBSP. They re-publish these changes for future service transactions.

2.2 Service-Oriented Software Architecture for E-Services

PSOM gives an overview of business service process virtualisation by means of software service abstractions. However, a more precise specification is needed in order to realise the indicated role functions of VBS transactions. Planning functions require a specification language that enables formal design, analysis and adjustment of all e-service parts. Control functions require a software architecture framework as well as executable languages to construct and run all e-service components. We provide a basis for both by means of a service-oriented metamodel for e-services that we refer to as *PARIS E-Service Metamodel (PSM).* Classes and relationships defined in the metamodel translate to components and relationships of a service-oriented software architecture for e-services as well as syntactical elements and semantics of an UML-based language for e-service design.

PSM is as agnostic as possible with respect to specific SOC standards and technologies. Our conception aims at clear abstraction layers of platform independent e-service models and platform specific software service models that are precisely mapped to conform to service virtualisation requirements. PSM architecture does not define technology platforms to implement platform specific service models. The rationale is to define a generic framework architecture that is applicable to multiple existing SOC standards and technologies and to uncouple architectural concepts from their fast evolution cycles. Concrete mappings need to be defined from platform specific software service models onto concrete ones.

In order to represent process organisation of service provision, PSM defines a process-driven SOA. Abstracting from specific technologies, PSM builds on a generic notion of *working process* [Kosiol 1962]. In organisation theory, working processes involve performers that carry out associated activities of structured methods for the sake of transforming inputs into outputs. In PSM, we adopt these generic concepts for both business service processes and software service interaction processes and we base their mapping upon these common roots. Technically, we use a workflow metamodel to formalise working processes and define metamodels of software service interaction and e-services as stepwise refinements.

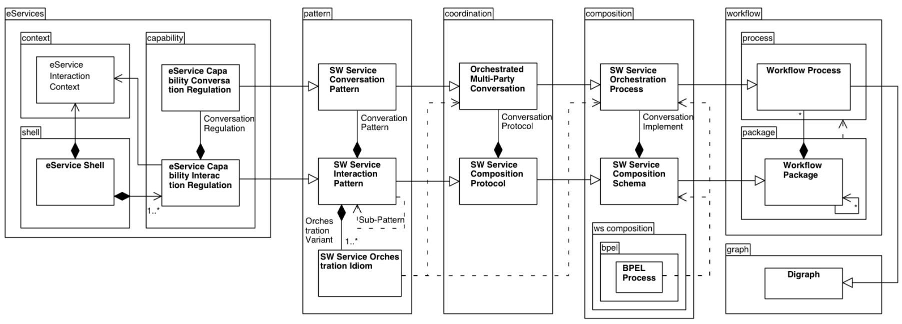
eServices
context
eService Interaction Context
capability
eService Capa bility Conversa tion Regulation
Conversation Regulation
shell
eService Shell
eService Capa bility Interac tion Regulation
1..*
pattern
SW Service Conversation Pattern
Converation Pattern
SW Service Interaction Pattern
Orches tration Variant
Sub-Pattern
1..*
SW Service Orches tration Idiom
coordination
Orchestrated Multi-Party Conversation
Conversation Protocol
SW Service Composition Protocol
composition
SW Service Orchestration Process
Conversation Implement
SW Service Composition Schema
ws composition
bpel
BPEL Process
workflow
process
Workflow Process
package
Workflow Package
graph
Digraph

Figure 2 – Overview of the PARIS E-Service Metamodel (PSM)

Fig.2 shows an overview of PSM in terms of UML packages and key classes. The foundation is a *graph metamodel* that introduces *directed graphs (Digraph)* as a basis for higher-level process descriptions. In turn, PSM defines a graph-based *workflow metamodel*. Its main ingredients are workflow process and package elements defined in respective sub-models. *Workflow processes* refine and extend digraphs as regards concepts of general working processes including participants, applications, data and structured activities. *Workflow packages* offer a further structuring principle that allows defining multiple workflow processes with a shared scope.

On the next level, PSM refines and extends workflow elements as software service interaction processes. They either serve as coordination protocols to regulate service interactions or as composition schemas for automated enforcement of these regulations. Composition schemas are defined in the *composition metamodel* for services. It refines and extends workflow processes as *software (SW) service orchestration processes*. They are structured within *software service composition schema* containers that refine workflow packages. Coordination protocols are defined in the *coordination* metamodel as generalised perspective on service interaction processes. Here, orchestration processes are refined as *orchestrated multi-party conversations* that focus on fewer elements of service interaction with respect to coordinative regulation. Conversations are aggregated into *software service composition protocols*, which represent observable choreography in the course of service composition.

Beyond schemas and protocols, PSM defines an abstraction of interaction process patterns to capture multiple variants of interaction that are equivalent in terms of functional effects but different in terms of non-functional properties. Interaction patterns are defined in the *pattern metamodel* for service interaction. It refines and extends orchestrated multi-party conversations as *software service conversation patterns*. Conversation patterns allow flexible non-deterministic description of orchestrated conversations that are part of interaction patterns. *Software service interaction patterns* refine composition protocols as non-deterministic choreography protocols. They aggregate sets of interaction sub-patterns whose conversation patterns together comprise typical interaction situations with multiple implementations. Each sub-pattern can be root of an individual interaction pattern. Interaction patterns also contain a set of *software service orchestration idioms* that each represent one possible transformation of non-deterministic service conversation patterns into deterministic orchestrated multi-party conversations and service orchestration processes.

On the topmost PSOM level, the *eService metamodel* defines concepts of VBSPs and relates them to software service interaction processes. It defines a SOA that realises assets, demands and capabilities. In particular, the metamodel refines software service conversation patterns as *eService capability conversation regulations* that capture provision logic of local or global virtual business service capabilities in a flexible non-deterministic way. eService capability conversation patterns are part of *eService capability interaction regulations* that are refined software service interaction patterns. eService capability interaction regulations specify interaction patterns that involve multiple capability conversations of sub-patterns and define alternative coordination protocols and orchestration schemas with varying non-functional properties. Brokers/providers in charge of a capability are responsible for its interaction regulation and all related conversation regulations of sub-patterns. Multiple capability interaction regulations are composed into an *eService shell* that represents a holistic VBSP. It also contains eService roles (brokers, providers, clients) and endpoints of involved assets and demands including message data formats that are defined within a common *eService interaction context*.

From a top-down perspective, PSM concepts define how the design of a VBSP is to be structured, how its structural parts are to be specified by means of interaction patterns and how these specifications are to be implemented by software service interfaces and composition schemas. The successive layers correspond to platform independent and specific parts of a modelling hierarchy that can be extended and mapped to various concrete software service technology platforms. We have defined one such refinement in terms of Web Service composition. Here, the *ws composition metamodel* defines extended concepts of Web Service architecture like WSDL-based interface descriptions. From this model, we have defined a mapping onto BPEL software service composition descriptions that can be deployed and executed within BPEL-compliant software service composition middleware platforms.

The PSM modelling hierarchy encourages stepwise development of e-services by means of structured design, verification and transformation methods. Such methods are at the heart of our e-service management approach as they allow meeting agility and flexibility requirements of VBS transactions. As a basis for such methods, PSM sub-models define elements of a graph-based process specification language for VBSPs and software service interaction processes [Zirpins and Emmerich 2008a].

4. RELATED WORK

The focus of this document is on utilising pattern-based, process-driven SOA for flexible modelling of service-oriented collaborative information systems that represent coordinative regulations of business service processes and allow for their rapid adaptation and enforcement. In terms of related work, we will first discuss examples of VO modelling approaches that build on service federation concepts and then look at exemplary approaches to flexible modelling of interaction-based software architecture for collaborative information systems.

Service federation emerged as a paradigm for modelling VO structures and realisation of ICT infrastructure by means of SOC [Camarinha-Matos 2005]. Exemplary projects adopting the service federation paradigm include *Fetish-ETF* [Camarinha-Matos et al. 2001]. It follows a peer-to-peer type VO approach for service providers of a tourism industry cluster. A *promoter node* utilises service market mechanisms for virtual enterprise initiation and promotes services either in *atomic* form or as *value added services* that build on general distributed business processes. A second example is the *WebBIS project* [Benatallah et al. 2000] that employs a service model with rule-based coordination concept. It introduces *virtual enterprise services* that specify and enforce their individual coordination logic by means of ECA rules. They group in *communities* that resemble a peer-to-peer type VO. Our approach shares the adoption of a service federation approach. It differs in its focus on virtual service enterprises and its adoption of pattern-based SOA for increased flexibility.

Pattern-based modelling of flexible software architecture is tackled by several other approaches. Pattern-based modelling of process-driven SOA is proposed by [Zdun et al. 2007]. They introduce *software patterns* to capture best practises and identify their building blocks as *pattern primitives* that can be used for individual SOA models. An approach for context-aware adaptive workflow applications is proposed by [Modafferi et al. 2006]. The approach builds on a methodology for development of *context-sensitive business processes*. It includes language primitives to model context-sensitive regions of a process by means of *context change patterns* and transformation rules to construct BPEL processes. Udupi and Singh introduce a modelling approach for service interaction that includes pattern concepts [Udupi and Singh 2008]. They focus on agreement and enactment of *service engagements* based on *commitments* of autonomous parties. Commitments are combined as (virtual) *organisations* that include policies of involved participants and determine their interactions. Specification is guided by design-patterns. The main distinguishing factor of our work is the federated characteristic of our service interaction patterns and in particular the structuring of interworking software service orchestrations.

5. SUMMARY AND OUTLOOK

In this paper we have introduced a SOA model for virtual service enterprises. In particular, we have proposed an UML-based SOA metamodel for realisation of e-services that represent virtual service processes. Planning of virtual service processes by means of e-service modelling provides means to regulate coordination of cooperative activities in a service network. Concepts of service interaction patterns allow for flexibility of coordinative regulations as well as their rapid enforcement

For demonstration and evaluation, we have conducted a case study experiment in the area of computational chemistry e-science. It is based on polymorph prediction research, where benefits were expected from virtualisation of e-science labs. In PARIS, we have examined virtualisation of polymorph prediction labs as virtual scientific service production networks [Zirpins and Emmerich 2008a].

Our current work concentrates on e-service management technology. We are adopting enterprise architecture to integrate e-service-management into the wider context service enterprises. The resulting framework defines an engineering approach to run virtual service enterprises by means of e-service technology. We are developing a service-oriented development methodology and implementing a tool set demonstrator for our e-service SOA that includes a model-driven development tool chain as well as an execution platform based on OGSA and BPEL.

7. REFERENCES

[Alonso et al. 2004] G. Alonso, F. Casati, H. Kuno, et al. *Web Services - Concepts, Architectures and Applications*. Springer, 2004.

[Benatallah et al. 2000] B. Benatallah, B. Medjahed, A. Bouguettaya, A. Elmagarmid, et al. Composing and Maintaining Web-based Virtual Enterprises. In *VLDB Workshop on Technologies for E-Services, Cairo, Egypt, September 2000*, pages 155–174. Informal Proceedings, 2000.

[Camarinha-Matos 2005] L. M. Camarinha-Matos. ICT Infrastructures for VO. In L. M. Camarinha-Matos, H. Afsarmanesh, and M. Ollus, editors, *Virtual Organisations: Systems and Practices*, pages 83–104. Springer, New York, 2005.

[Camarinha-Matos et al. 2001] L. M. Camarinha-Matos, H. Afsarmanesh, E. C. Kaletas, and T. Cardoso. Service Federation in Virtual Organizations. In George L. Kovács, Peter Bertók, and Géza Haidegger, editors, *Digital Enterprise Challenges: Life-Cycle Approach to Management and Production*, IFIP Conference Proceedings 205, pages 305–324. Kluwer, 2001.

[Kosiol 1962] E. Kosiol. *Organisation der Unternehmung.* , Wiesbaden, 1962.

[Modafferi et al. 2006] S. Modafferi, B. Benatallah, F. Casati, and B. Pernici. A methodology for designing and managing context-aware workflows. In *Mobile Information Systems II; IFIP International Working Conference on Mobile Information Systems, (MOBIS) Leeds, UK, December 6–7, 2005*, volume 191/2005 of *IFIP*, pages 91–106. Springer Boston, 2006.

[Papazoglou and Georgakopoulos 2003] M. P. Papazoglou and D. Georgakopoulos. Service-Oriented Computing: Introduction. *Communications of the ACM*, 46(10):24–28, 2003.

[Picot and Neuburger 1998] A. Picot and R. Neuburger. Virtuelle Organisationsformen im Dienstleistungssektor. In M. Bruhn and H. Meffert, editors, *Handbuch Dienstleistungsmanagement*, pages 513–534. Gabler, Wiesbaden, 1998.

[Udupi and Singh 2008] Y. B. Udupi and M. P. Singh. Design Patterns for Policy-Based Service Engagements. Technical Report TR-2008-3, Department of Computer Science, North Carolina State University, 2008.

[Zdun et al. 2007] U. Zdun, C. Hentrich, and S. Dustdar. Modelling Process-Driven and Service-Oriented Architectures Using Patterns and Pattern Primitives. *ACM Transaction on the Web*, 1(3):14:1–14:44, 2007.

[Zirpins and Emmerich 2008a] C. Zirpins and W. Emmerich. Service-Oriented Modelling and Design of Virtual Business Services. Research notes, University College London, Dept. of Computer Science, 2008.

[Zirpins and Emmerich 2008b] C. Zirpins and W. Emmerich. A Reference Model of Virtual Service Production Networks. *Service Oriented Computing and Applications*, (Special Issue on: Service Intelligence and Service Science), 2008.