Service Co-operation Patterns and their Customised Coordination

Christian Zirpins[1], Winfried Lamersdorf[1]

[1] Distributed Systems Group- VSIS
University of Hamburg, Hamburg (Germany)
{zirpins, lamersdorf}@informatik.uni-hamburg.de

Abstract. Service Oriented Computing is meant to support loose relationships between organisations. Such relationships constitute co-operation procedures that translates to interaction processes via Webservices. Service composition deals with the specification and automated enforcement of such interaction processes and its predominant approach is orchestration, where a workflow management system (WFMS) is proactively coordinating the interaction activities. In most cases, the orchestration process is regarded as an implicit result of co-operation logic (actually, they are often the same) but the reverse impact of operational coordination on co-operation logic are often neglected. In this position paper, we claim that the choice of coordination alternatives impacts the quality of service and has to be customised to actual service cases and their individual participants. We will introduce a potential solution approach that revolves around service co-operation patterns. We borrow the paradigms of patterns/idioms that are well known from object-oriented design/development and apply it to co-operation procedures and orchestration processes. This approach allows studying a) reusable co-operation patterns typical for service relationships and b) for each pattern a range of possible coordination idioms. We sketch a technique that is intended to refine a composition process based on an analysis of its co-operation patterns and the application of suitable coordination idioms selected by rules in terms of the service context.

1 Introduction

Webservices are software components that provide self-contained functionality over Internet-enabled, interoperable interfaces and publish a common description of their characteristics to be dynamically discovered, selected and accessed by clients. They provide fundamental building blocks for *Service Oriented Computing (SoC)* [1] that aims to support service relationships between organisational participants. However, a single Webservice is almost never capable of representing a complete *application-level service* (e.g. a flight booking service). On the one hand, even a basic application service normally includes a non-trivial *bilateral co-operation procedure (CoP)* between client and provider (e.g. book → encash) that includes co-operation functions on both sides and clear *conversational logic*. On the other hand, an application service typically splits into functional parts (search flight offers, book flights) of multiple providers (e.g. flight broker, airline) and includes their *composition logic*, resembling a *multilateral CoP*.

In either case, the field of *Webservice Composition* provides concepts and techniques to assemble basic functional Webservices into composite services that constitute a considerable step towards application services. In particular, service composition is concerned about the coordination of composite services by means of *service orchestration processes (SOP)*. Orchestration languages like BPEL4WS adopt concepts of *workflow (Wf)* (i.e. automated collaborative working processes) to specify flows of control and data between Webservice operations. As SoC focuses multi-organisational relationships, composition typically includes multiple interconnected SOPs controlled by different participants. Therefore, *cross-organisational workflow (CoWf)* is an area that is closely related to service composition. It focuses Wfs that span multiple organisational domains. The central problem is the decomposition of single Wfs with respect to the set of participating organisations. Facets of this problem comprise meta-models and modelling (e.g. [2]), verifiable consistency (e.g. [3]), enforcement and runtime architecture (e.g. [4]).

As a valid straight forward solution for service realisation, several approaches like eFlow [5], SELF-SERV [6], DySCo [7] or FReSCo [8] mapped CoP to SOPs/CoWfs (anticipate Fig.4 for an overview). However, it has to be minded that CoP and CoWf differ in some subtle aspects like change frequency of participants and additional facets of their interrelation [9]. In this paper we propose to address one such facet concerned with coordination: CoWf-based CoP implies a fixed decision on operational coordination (i.e. decomposition, refinement and distribution of local Wfs). This is not desirable for services as it impacts their characteristics and should be rather treated as a separated aspect that can be decided on dynamically and independently from the core service logic. Our solution involves *coordination idioms* that get dynamically selected and applied to *co-operation patterns* in the CoP by *rules* based on the actual service context.

The rest of this position paper outlines the facets of our envisioned service coordination concept. Sect. 2 details the impacts of coordinative variations for composite services. Sect. 3 proposes patterns and rules as an approach to achieve flexibility of service coordination. Sect. 4 concludes.

2 Coordination Choices for Service Co-operation Procedures

As we saw, process-oriented design together with Wf-based realisation is a major approach to service composition. While it has merits (e.g. it is rather intuitive), it also has drawbacks, some of them laying in the nature of Wf itself and some of them originating from the different requirements of CoWf in contrast to SoC. A particular source of problems revolves around coordination

We distinguish *logical dependencies* that are used to model CoP logic from *operational coordination* as the actual procedure of enforcement. CoWf processes represent logical dependencies and simultaneously act as instruction for their operational coordination (also process engines usually transform Wf processes into a representation that is optimized for enforcement; e.g. ECA rules for active DBMS). However, coordination merely emerges implicitly as a side-effect of the dependency model and not explicitly for good reasons. Actually, there are usually multiple choices of enforcement. They result from partitioning the set of dependencies and delegating their operational coordination to the involved organisations. This is crucial, because the actual choice of a coordination structure affects certain characteristics of the CoWf. Colombo, Francalanci and Pernici [10] describe the effect in terms of *organisational structure*. We reckon an impact on additional *non-functional characteristics* that affect the *quality of service (QoS)* as will be indicated in the following example.

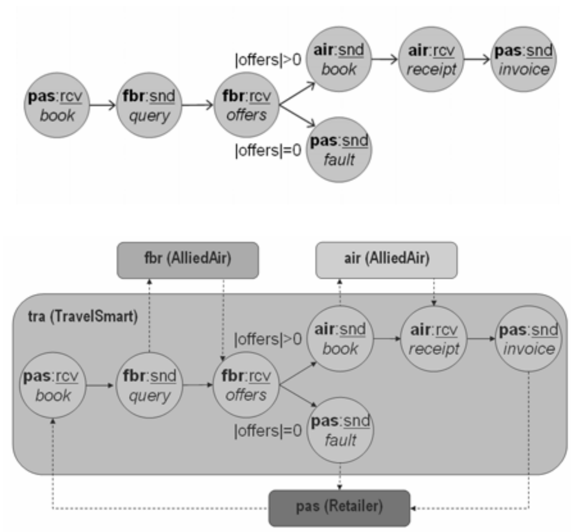

Fig. 1. Flight Booking Service with Basic Coordination Choice: a) Co-operation Procedure (upper), b) Central Orchestration Process (lower)

Fig.1a shows a very simple flight booking CoP that contains *interactions* (circles) and *transitions* (arrows). Interactions represent invocations of co-operation functions (e.g. *book*) either originating from (rcv) or going to (snd) a *role* that represents a participant (**tra** = travel agency, **pas** = passenger, **air** = airline, **fbr** = flight broker). The example informally says that a passenger will access a booking service and a flight broker will be queried for an airline to forward the request to. If an offer is found, a booking request will be issued to the airline and after its confirmation, an invoice will be sent to the passenger. Otherwise, a fault will be reported.

Up to now, this CoP tells nothing about the participant(s) that will enforce its dependencies. Fig.1b and fig.2 show three alternative refinements in terms of operational coordination. For each, the CoP is portioned into interrelated SOPs that are assigned to roles. Co-operation functions translate to Webservice operations.

The example shows that SOPs can vary while the logical co-operation stays exactly the same. However, coordination choices affect various non-functional characteristics. On the one hand, differences emerge on system level. E.g., fig.1b features central coordination that results in lower overhead (and thus reduced costs) but might suffer from common drawbacks like emergence of a bottleneck. Fig.2 features distributed coordination that leads to more complexity but allow for opportunities of local optimisations and increased parallelism.

On the other hand, differences appear on application level. While fig.1b empowers TravelSmart to coordinate (and control) the whole CoP, fig.2a delegates a certain level of coordination and control to other participants. This results in an explicit shift of rights and responsibilities as well as an implicit shift of autonomy. Such differences may be crucial to specific business constraints of an organisation. While it is in some cases acceptable (e.g. fig.2a: TravelSmart delegates short-term consumers to public accessible sub-services because customer retention is considered unimportant) it is not acceptable in others (e.g. fig.1b: TravelSmart acts as an integrator of non-public sub-services for a retailer to which TravelSmart wants to keep an exclusive long-term relationship). However, it is possible to partly compensate such affects by refining the operational co-operation (e.g. fig.2b: in spite of delegated coordination, TravelSmart regains control as well as the direct and exclusive relationship to its customer).

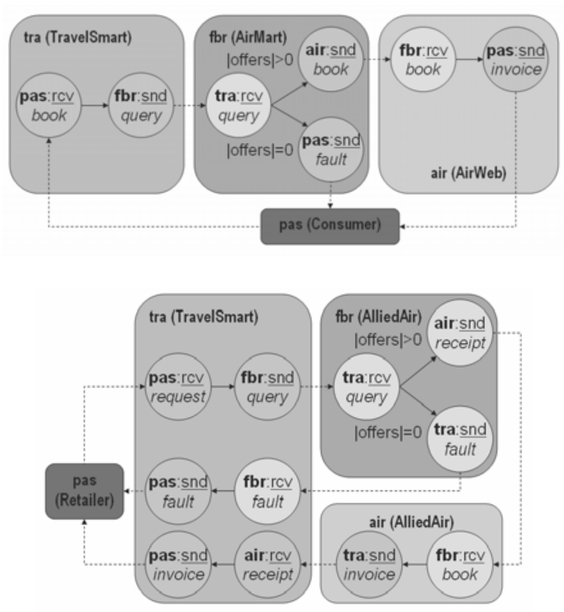

Fig. 2. Distributed Coordination Choices for the Flight Booking Service: a) Decentral Orchestration, b) Changed Control (upper), Decentral Orchestration, Preserved Control (lower)

In the end, the choice of coordination depends on requirements of the participants. While participants are rather fixed in CoWf, they frequently change for services. Thus, the choice of service coordination should be open until the participants are known. Then, the composition logic should be refined to satisfy their QoS requirements. In the following section, we propose the use of design patterns to capture typical structures of generic service CoP and associated implementation patterns (or idioms) to capture their pre-analysed coordination choices.

3 Investigating Service Patterns and their Rule Based Refinement

Before we outline the ideas of our approach, we summarize the challenges that emerge from last section's observations in the context of the *service composition lifecycle* proposed by Yang and Papazoglou [11]. Such a lifecycle is structured into five phases: *1) planning* (synthesis of service logic), *2) definition* (abstraction of service composition), *3) scheduling* (analysis of possible composition refinements in the context of a new service case), *4) construction* (assembly of concrete composition for the case) and *5) execution* (enforcement of concrete composition).

In this work, we suppose service logic (i.e. conversation- and composition logic) as external input (e.g. from business process modelling) and do not directly interfere with its synthesis. Thus, our methodology can be classified as *semi-fixed composition* [11] and considers phases 2-5, where we face the following problems:

- *Lp2) Definition phase:* specify coordination-independent CoP. A *CoP meta-model* (CoPMm) is needed that can represent given conversation/composition logic in terms of dependencies between abstract service components (co-operation functions) and abstract participants (roles) and must not imply any constraints for operational coordination. Clearly, some form of *high level abstractions* is needed to support the design of CoPs on the basis of a possibly informal description of service logic.
- *Lp3) Scheduling phase:* analysis and evaluation of coordination choices. Exact information is needed about a) the range of possible coordination choices for a given CoP b) the range of relevant characteristics and the affects of individual choices on them c) the service case's context, that is, the group of possible participants together with their characteristics and requirements. Analysis has to consider all this information and prepare it for evaluation. This requires pre-defined metric of qualitative measurements and a formal framework for automated arguing.
- *Lp4) Construction phase*: refinement of CoPs into SOPs. Model transformation capabilities are needed from the CoPMm to a CoWf Mm of choice. Additionally, structural transformations of CoWfs are needed to refine CoP dependencies according to the choice of operational co-operation. This doesn't only require the capability to merely do any such transformation but exact knowledge how to realise each specific choice of co-operation.
- *Lp5) Execution phase*: enforcement of SOPs. As SOPs resemble standard CoWfs, there are no specific problems in terms of this discussion.

To cope with this family of coordination related problems, an approach is generally required to manage the dependencies between high-level service co-operation and low-level Wf coordination. In particular, it has to a) free the conceptual level from unintended implications on operational level and b) leverage relevant implications, which emerge from choices on operational level, up onto conceptual level. While a) is a question of technique (e.g. reasoning capabilities, model transformation), b) requires mature knowledge of a broad variety of scenarios (what co-operation choices are there and for which forms of co-operation do they carry? What are the implications of specific coordination choices in terms of which service characteristics?).

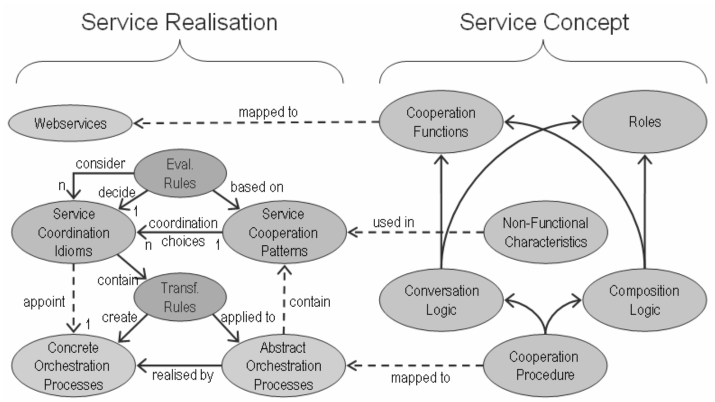

Fig. 3. Conceptual Overview

As such an approach, we propose a framework that contains both aspects. The technical part of this framework contains three building blocks: *service co-operation patterns (SCP)*, *service coordination idioms (SCI)* and *service refinement rules (SRR)*. SCPs are meant to abstract forms of co-operation in CoP (e.g. brokerage, intermediation, delegation…). This includes a *pattern meta-model* as well as a *cooperation catalogue* containing a collection of *pre-defined patterns*. They are intended as building blocks for CoP design in lp2 and as knowledge base for CoP analysis in lp3. The idea behind an SCI is to represent a cooperation choice for an SCP. A *coordination catalogue* contains *pre-defined idioms* and each SCP is associated with a specific range of them. An SCI includes the requirements and implications of its application in terms of the SCP's characteristics as well as a generic implementation (see STR below) in terms of a CoWf model. SCIs are intended as a knowledge base for analysis and evaluation in lp3. Additionally they drive the creation of SOPs in lp4. Finally, the intent of SRRs is to link everything together. Essentially, we aim to adopt a rule-based approach for evaluation in lp3 and transformations in lp4. *Evaluation rules (SER)* represent application knowledge (i.e. business rules) to find and decide choices of cooperation. *Transformation rules (STR)* change the representation and structure of Wfs and are part of SCIs generic implementation mechanism. Fig.3 gives an overview of all concepts and their associations.

Beyond the technical challenge to define meta-models for patterns and idioms, a system of rules and mechanisms for process analysis and transformation, the conceptual challenge is to identify a range of real patterns, idioms and rules for close examination:

- A range of typical and/or useful co-operation situations for inter-organisational service relationships has to be informally identified in an intuitive way (e.g. based on scenarios).

- For each co-operation situation, an examination and specification has to be done in terms of a) participating roles b) detailed interaction dependencies c) relevant non-functional characteristics
- For each pattern, the range of possible coordination choices has to be informally identified with a systematic method (that is, *all* possible choices have to be considered).
- For each coordination choice, an examination and specification has to be done in terms of a) prerequisites b) impact on pattern characteristics c) application context
- For each pattern and its idioms, a scenario-based examination of rules can be done to verify the concepts and describe their relationships.

The rationale for these investigations is to gain solid knowledge about the coordination of service co-operation. This knowledge is provided in a reusable form. Concrete patterns and idioms become part of the system's knowledge base and can be applied in the lifecycle. The existence of such knowledge is crucial for the application of our approach and forms the conceptual part of our framework. Its investigation constitutes an important and distinctive part of our research.

3 Related Work

Our approach is placed in the field of workflow-based service composition (see sec.1 for general work in this area) and copes with coordination problems that are partly rooted in workflow itself and partly originate from the different requirements of service composition (see sec.2 for related work on this). In particular we adopt techniques of patterns and rule based transformation in the context of workflow.

The use of *rules in workflow management* is quite common. Apart from the integration of rules as elements inside workflows, rules have been used on meta-level for *workflow adaptation* [12, 13]. In this case, rules govern modifications that are applied to a workflow either statically at design-time or dynamically at runtime to add flexibility. Recently, business rules have been proposed to construct BPEL4WS service composition processes [14]. This approach has particular similarities to ours in terms of its use of rules to conduct different transformations of SOPs during service composition lifecycle. Though, it does not consider pattern mechanisms or investigate concrete process structures.

Patterns provide means to conserve and reuse knowledge about the solution of a generic problem. They range from informal guidelines used in system design (*design patterns*) to customizable code fragments (*implementation patterns* or *idioms*). In particular, patterns are widely known for their use in object-oriented design and architecture [15]. Pattern concepts have also been applied to workflow management. There have been foundational studies on basic control-flow structures named *workflow patterns* [16] that can be used to examine and compare general workflow languages. In [17] a formal model is proposed for rule idioms that can be instantiated as rule elements in workflow schemas in the Chimera-Exc language. In [10], high-level design patterns for organisational coordination and control structures are proposed and related to corresponding CoWf.

Also, in recent literature, pattern-based approaches have been proposed for service composition. The authors of [18, 19] propose architectural design patterns that give good indications on starting points for our investigation of CoP and SOP patterns. Another proposal is to use design patterns of service composition logic [20]. To the best of our knowledge, work on process patterns for service co-operation or coordination has not been published yet.

4 Conclusion

In the emerging research field of service oriented computing and especially in service composition, many approaches are closely related to workflow concepts. In particular, concepts of cross-organisational workflow are often used to model and execute composite services. However, first doubts appear about the appropriateness of workflow concepts for service composition as the later is believed to imply more complexity, more dynamics and more facets in the relation of participants. Such doubts are encouraged by findings from CoWf research that indicate shortcomings of current concepts as regards the support of coordination aspects. The problem is that most efforts concentrate on developing generic techniques to solve problems of an application area that is generally not well understood yet. Only few approaches aim to investigate concrete characteristics (specific classes of problems, their requirements and solution strategies) of composite services.

In this position paper we propose to investigate the specific facet of coordination aspects in service composition. We gave indications about the relevance of coordination choices for the enforcement of service composition dependencies as those choices rebound on service characteristics. To get a grip on this phenomenon, we proposed generic mechanisms that allow representing relationships (as rules: SRR) between concrete service composition logic (as patterns: SCP) and their concrete coordination choices (as idioms: SCI). The ability to model patterns of composition logic and their idioms of coordination will enable us to formulate and structure knowledge about a range of concrete problems and solutions of service composition that we intent to examine. Ultimately, the generic mechanisms together with the concrete knowledge translate into a framework to support the lifecycle of service composition.

References

1. Papazoglou, M.P. and D. Georgakopoulos, *Service-oriented computing: Introduction.* Communications of the ACM, 2003. **46**(10): p. 24-28.
2. Aalst, W.M.P.v.d. and M. Weske, *The P2P Approach to Interorganizational Workflows*, in *Advanced Information Systems Engineering, 13th International Conference, CAiSE 2001, Interlaken, Switzerland, June 4-8, 2001, Proceedings*, K.R. Dittrich, A. Geppert, and M.C. Norrie, Editors. 2001, Springer. p. 140-156.

3. Mecella, M., B. Pernici, and P. Craca, *Compatibility of e -Services in a Cooperative Multi-platform Environment*, in *Technologies for E-Services: Second International Workshop TES 2001, Rome, Italy, September 14-15, 2001, Proceedings*, F. Casati, D. Georgakopoulos, and M.-C. Shan, Editors. 2001, Springer. p. 44-57.
4. Aalst, W.M.P.v.d., *Process-oriented architectures for electronic commerce and interorganizational workflow.* Information Systems, 1999. **24**(8): p. 639-71.
5. Casati, F. and Ming Chien Shan, *Dynamic and adaptive composition of e-services.* Information Systems, 2001. **26**(3): p. 143-63.
6. Dumas, M., B. Benatallah, and Z. Maamar, *Definition and Execution of Composite Web Services: The SELF-SERV Project.* Data Engineering Bulletin, 2002. **25**(4).
7. Piccinelli, G., A. Finkelstein, and S.L. Williams, *Service-oriented work-flows: the DySCo framework*, in *Proc. 29th Euromicro Conference, Antalya, Turkey*. 2003.
8. Zirpins, C., W. Lamersdorf, and G. Piccinelli, *A Service Oriented Approach to Inter-organisational Cooperation*, in *Proc. IFIP International Conference on Digital Communities in a Networked Society: eCommerce, eBusiness, and eGovernment (I3E) 2003*, M. Mendes, R. Suomi, and C. Passos, Editors. 2004, Kluwer Academic Publishers.
9. Yang, J., W.-J.v.d. Heuvel, and M.P. Papazoglou, *Tackling the Challenges of Service Composition in e-Marketplaces*, in *Proceedings of the 12th International Workshop on Research Issues on Data Engineering: Engineering E-Commerce/E-Business Systems (RIDE-2EC 2002), San Jose, CA, USA*. 2002.
10. Colombo, E., C. Francalanci, and B. Pernici, *Modeling Coordination and Control in Cross-Organizational Workflows*, in *Proc. CoopIS/DOA/ODBASE 2002*, R. Meersmann and Z. Tari, Editors. 2002, Springer. p. 91 ff.
11. Yang, J. and M.P. Papazoglou, *Service Components for Managing Service Composition Lifecycle.* Information Systems, 2004. **29**(2): p. 97-125.
12. Joeris, G. and O. Herzog, *Managing Evolving Workflow Specications with Schema Versioning and Migration Rules*. 1999, University of Bremen.
13. Zirpins, C. and G. Piccinelli, *Evolution of Service Processes by Rule Based Transformation*, in *IFIP International Conference on Digital Communities in a Networked Society: eCommerce, eBusiness, and eGovernment (I3E) 2004*. 2004 (to appear).
14. Orriëns, B., J. Yang, and M.P. Papazoglou, *A Framework for Business Rule Driven Service Composition*, in *Technologies for E-Services, 4th International Workshop, TES 2003, Berlin, Germany, September 8, 2003, Proceedings*, B. Benatallah and M.-C. Shan, Editors. 2003, Springer. p. 14-27.
15. Buschmann, F., et al., *Pattern-Oriented Software Architecture - A System of Patterns*. 1996: Wiley and Sons Ltd.
16. Aalst, W.M.P.v.d., et al., *Workflow Patterns.* Distributed and Parallel Databases, 2003. **14**(3): p. 5-51.
17. Casati, F., et al., *Using Patterns to Design Rules in Workflows.* IEEE Trans. Software Eng., 2000. **26**(8): p. 760-785.
18. Rana, O.F. and D.W. Walker, *Service Design Patterns for Computational Grids*, in *Patterns and Skeletons for Parallel and Distributed Computing*, F.A. Rabhi and S. Gorlatch, Editors. 2003, Springer: London. p. 237-264.
19. Benatallah, B., et al., *Towards Patterns of Web Services Composition*, in *Patterns and Skeletons for Parallel and Distributed Computing*, F.A. Rabhi and S. Gorlatch, Editors. 2003, Springer: London. p. 265-296.
20. Edmond, D. and M.T. Tut, *The Use of Patterns in Service Composition*, in *proc. CAISE'02*. 2002.