Grant Agreement N° 215483

Title:	*CD-JRA-2.3.4: Decision support for local adaptation*
Authors:	*CNR, INRIA, SZTAKI, TUW*
Editor:	*Zsolt Nemeth (SZTAKI)*
Reviewers:	*Andreas Gehlert (UniDue)*
	Marco Pistore (FBK)
	Dimka Karastoyanova (USTUTT)
	Jean-Louis Pazat (INRIA)
Identifier:	*Deliverable #CD-JRA-2.3.4*
Type:	*Deliverable*
Version:	*1.0*
Date:	*15 December, 2009*
Status:	*Final*
Class:	*External*

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement n° 215483 (S-Cube).

File name: cover.doc

Grant Agreement N° 215483

Management Summary

This deliverable is aimed at summarizing the joint research in WP-JRA-2.3. related to decision support for local adaptation. It is an intermediate stage on the research roadmap, starting from issues of local adaptation and self-healing (CD-JRA-2.3.2) to the most complex case involving distributed multi-level adaptation (CD-JRA-2.3.8), where we investigate and integrate certain methods and techniques incrementally. The work is based on and motivated by the antecedent deliverable "Basic requirements for self-healing services and decision support for local adaptation" (CD-JRA-2.3.2) and is focused on local adaptation and decision which we consider one of the most important ways to investigate the applicability of certain policies to trigger local adaptation mechanism, and is organized around the general adaptation framework introduced in CD-JRA-2.3.2. Results are presented in 10 published papers that constitute the core contribution of this deliverable. The work is positioned within the Integrated Research Framework (IRF, WP-IA-3.1), internal WP-JRA-2.3 research architecture and overall WP-JRA-2.3 goals and visions.

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement n° 215483 (S-Cube).

Members of the S-Cube consortium:

University of Duisburg-Essen (Coordinator)	Germany
Tilburg University	Netherlands
City University London	U.K.
Consiglio Nazionale delle Ricerche	Italy
Center for Scientific and Technological Research	Italy
The French National Institute for Research in Computer Science and Control	France
Lero - The Irish Software Engineering Research Centre	Ireland
Politecnico di Milano	Italy
MTA SZTAKI – Computer and Automation Research Institute	Hungary
Vienna University of Technology	Austria
Université Claude Bernard Lyon	France
University of Crete	Greece
Universidad Politécnica de Madrid	Spain
University of Stuttgart	Germany
University of Hamburg	Germany
Vrije Universiteit Amsterdam	The Netherlands

Published S-Cube documents

All public S-Cube deliverables are available from the S-Cube Web Portal at the following URL:

http://www.s-cube-network.eu/results/deliverables/

The S-Cube Deliverable Series

Vision and Objectives of S-Cube

The Software Services and Systems Network (S-Cube) will establish a unified, multidisciplinary, vibrant research community which will enable Europe to lead the software-services revolution, helping shape the software-service based Internet which is the backbone of our future interactive society.

By integrating diverse research communities, S-Cube intends to achieve world-wide scientific excellence in a field that is critical for European competitiveness. S-Cube will accomplish its aims by meeting the following objectives:

- Re-aligning, re-shaping and integrating research agendas of key European players from diverse research areas and by synthesizing and integrating diversified knowledge, thereby establishing a long-lasting foundation for steering research and for achieving innovation at the highest level.
- Inaugurating a Europe-wide common program of education and training for researchers and industry thereby creating a common culture that will have a profound impact on the future of the field.
- Establishing a pro-active mobility plan to enable cross-fertilisation and thereby fostering the integration of research communities and the establishment of a common software services research culture.
- Establishing trust relationships with industry via European Technology Platforms (specifically NESSI) to achieve a catalytic effect in shaping European research, strengthening industrial competitiveness and addressing main societal challenges.
- Defining a broader research vision and perspective that will shape the software-service based Internet of the future and will accelerate economic growth and improve the living conditions of European citizens.

S-Cube will produce an integrated research community of international reputation and acclaim that will help define the future shape of the field of software services which is of critical for European competitiveness. S-Cube will provide service engineering methodologies which facilitate the development, deployment and adjustment of sophisticated hybrid service-based systems that cannot be addressed with today’s limited software engineering approaches. S-Cube will further introduce an advanced training program for researchers and practitioners. Finally, S-Cube intends to bring strategic added value to European industry by using industry best-practice models and by implementing research results into pilot business cases and prototype systems.

S-CUBE materials are available from URL: http://www.s-cube-network.eu/

Contents

1 Foreword

According to the Description of Work (Grant Agreement for: Network of Excellence Annex I Description of Work) this deliverable is aimed at documenting how a service can be adaptable and self-aware while adhering to a set of given specifications and mechanisms and it will investigate the applicability of certain policies to trigger local adaptation mechanism.

The joint research work in WP-JRA-2.3 is composed of two tasks: T-JRA-2.3.1: Infrastructure Mechanisms for the Run-Time Adaptation of Services and T-JRA-2.3.2 Service Registration and Search. The current deliverable is related to T-JRA-2.3.1 and aimed at summarizing the investigations related to decision support for local adaptation. To achieve this goal, work is based on and motivated by the antecedent deliverable "Basic requirements for self-healing services and decision support for local adaptation" (CD-JRA-2.3.2) and prepares the way to the forthcoming deliverables "Specifications of policies and strategies for local adaptation" (CD-JRA-2.3.6) and "Specifications of policies and strategies for distributed and multi-level adaptation" (CD-JRA-2.3.8.) On the other hand T-JRA-2.3.2 proceeds with investigation of service registries and discovery methods; provider evaluation and rankings. These works are not derived from the requirements set in CD-JRA-2.3.2 but they are strongly related and interwoven with issues of T-JRA-2.3.1 and hence, some of relevant results are reported here.

The joint work is focused on local adaptation and decision which we consider one of the most important ways to investigate the applicability of certain policies to trigger local adaptation mechanism, and is organized around the general adaptation framework introduced in CD-JRA-2.3.2 [11] and summarized in Section 2.3. Results of the research work are presented in 10 published papers that constitute the core contribution of this deliverable. The work is positioned within the Integrated Research Framework (IRF, WP-IA-3.1), internal WP-JRA-2.3 research architecture and overall WP-JRA-2.3 goals and visions.

2 Introduction

As it was stated in CD-JRA-1.2.1 [13]: "The dynamic nature of the business world highlights the continuous pressure to reduce expenses, to increase revenues, to generate profits, and to remain competitive. This requires Web services to be highly reactive and adaptive. It should be equipped with mechanisms to ensure that their constituent component Web services are able to adapt to meet changing requirements. In fact, services are subject to constant changes and variations. Services can evolve due to changes in structures (attributes and operations), in behavior (when services are interacting) and policies." There is a strong consensus on the fact that services and service infrastructures must be able to perform (self-) adaptation which is the topic of WP-JRA-2.3. We set up our research roadmap starting from issues of local adaptation and self-healing (CD-JRA-2.3.2) to the most complex one involving distributed multi-level adaptation (CD-JRA-2.3.8) and investigating and integrating certain methods and techniques incrementally. The current stage reported in this deliverable comprises investigating decision support for local adaptation and the related techniques.

The aspects of (local) adaptation have been studied in CD-JRA-2.3.2 [11], basic definitions, discussions and requirements can be found there. On the lowest level one considers adaptation of one service on its own. The second level considers adaptation between services in a composition to satisfy the needs of an application. On the highest level we investigate the adaptation of several applications, running in parallel, each application being a composition of services itself. The requirements for the first level of adaptation,local adaptation have been described in [11]. Roughly, such an adaptation framework should contain at least four functionalities (albeit not necessarily implemented as four independent components in an architectural point of view): monitoring, decision making, action planning and execution. All other requirements are tied to these main functionalities. The aim of the current deliverable is to study decision support mechanisms for local adaptation - albeit, as we will see, decision support hardly ever appears in a standalone way, rather in strong functional and/or architectural cohesion with other components.

This deliverable "Decision support for local adaptation" has some of its foundations in CD-JRA-2.3.2 "Basic requirements for self-healing services and decision support for local adaptation" [11]. The work presented in this deliverable was organized according to the outputs and observations of the antecedent deliverable by distilling the most important research aspects that we followed down. This deliverable is also a prerequisite for CD-JRA-2.3.6 "Specifications of policies and strategies for local adaptation" as well as CD-JRA-2.3.8 "Specifications of policies and strategies for distributed and multi-level adaptation" hence, work was also driven towards establishing a basis for future investigations in these topics.

The deliverable is composed of published papers that are the outcome of joint WP-JRA-2.3 work summarizing aspects of local adaptation with a special focus on decision support. Section 2.1 ties the work to the Integrated research Framework whereas Section 2.2 positions the research work in the WP-JRA-2.3 research vision. Section 2.3 summarizes the main concepts of local adaptation. Section 3 introduces the papers, their main contributions, their relationship to the topic and the overall goals of the WP as well as their contribution to other WPs. The papers themselves are attached as Appendices.

2.1 Connections to the Integrated Research Framework

The Integrated Research Framework (IRF) [1] defines four views (other aspects of the IRF, like research challenges, questions and results are being developed simultaneously with this deliverable and omitted.)

The *Conceptual Research Framework* organizes the joint research activities by providing a high-level conceptual architecture for the principles and methods for engineering service based applications, as well as for the technologies and mechanisms which are used to realize those applications. The work presented in this deliverable is clearly related to the 'Service Infrastructure' domain in the horizontal classification whereas addresses issues of 'Adaptation, Monitoring' and 'Quality Definition, Negotiation and Assurance' of the cross-cutting vertical categorization.

The *Reference life-cycle* complements the static view of the conceptual research framework. It is composed of two main cycles: one corresponds to the classical application design, deployment and provisioning; one corresponds to the adaptation perspectives. The work presented in this deliverable corresponds to the adaptation cycle by investigating mechanisms that detect problems, changes, and needs for adaptation; identify possible adaptation strategies; and enact them. There is a special focus in our work on the latter issue: deployment and provisioning of the modified service.

The *Logical Run-Time Architecture* unifies all runtime mechanisms into a coherent framework. The work reported in this deliverable is related to nearly all components of the logical architecture: Monitoring Engine, Adaptation Engine, Negotiation Engine, Runtime QA Engine and Resource Broker.

The *Logical Design Environment* is complementary to the run-time architecture and its purpose is to provide a context where to place the envisioned techniques and mechanisms that support the analyst and designer in the design of a SBA. Our work is less related to modeling and verification, most contributions are related to the 'Transformation and Generation' and 'Deployment', especially their infrastructure related layers.

2.2 Connections to the JRA-WP-2.3 research architecture

Research work in WP-JRA-2.3 is driven by the Work Package vision that structures the research work internally. Figure 1 illustrates the overall research architecture of WP-JRA-2.3: research on service infrastructures is comprised in three threads, Service Discovery, Service Registries and Service Execution. Orthogonally different approaches are separated in three layers.

– Service Discovery Thread (A) - Service discovery is a fundamental element of service-oriented architectures, services heavily rely on it to enable the execution of service-based applications. Novel discovery mechanisms must be able to deal with millions of services. Additionally, these discovery mechanisms need to consider new constraints, which are not prevalent today, such as Quality of Experience requirements and expectations of users, geographical constraints, pricing and contractual issues, or invocability.

– Service Registry Research Thread (B) - Service registries are tools for the implementation of loosely-coupled service-based systems. The next generation of registries for Internet-scale service ecosystems are emerging, where fault tolerance and scalability of registries is of eminent importance. Autonomic registries need to be able to form loose federations, which are able to work in spite of heavy load or faults. Additionally, a richer set of metadata is needed in order to capture novel aspects such as self-adaptation, user feedback evaluation, or Internet-scale process discovery. Another research topic is the dissemination of metadata: the distributed and heterogeneous nature of these ecosystems asks for new dissemination methods between physically and logically disjoint registry entities, which work in spite of missing, untrusted, inconsistent and wrong metadata.

– Runtime Environment Research Thread (C) - There is an obvious need for automatic, autonomic approaches at run-time. As opposed to current approaches we envision an infrastructure that is able to adapt autonomously and dynamically to changing conditions. Such adaptation should be supported by past experience, should be able to take into consideration a complex set of conditions and their correlations, act proactively to avoid problems before they can occur and have a long lasting, stabilizing effect.

In alignment with the lifecycle aspect of the Integrated Research Framework, the presented work – due to its nature –, is related mostly to runtime activities hence, to research topics in thread C in Figure 1. Topic C2 is directly covered, C1 has also presented progress whereas C3 represents mostly future work. Service registry and discovery form an integral part of any adaptation related techniques hence, some aspects of A1, B2 and B3 are covered. The nature inspired model addresses adaptation and self-* from a high level point of view and is related to level 1 of policies (horizontal layer of the research architecture) involving (but not specializing in) A1, B1, C1.

2.3 Concepts of local adaptation

To present and position the joint research work, we briefly summarize the conceptual framework that was used for setting up the requirements for local adaptation and self-healing [11]. The adaptation framework (Figure 2) follows the functional decomposition of an autonomic manager suggested in [17], classifying the adaptation in *monitoring*, *analyzing*, *planning* and *executing* parts. Each part triggering the next one: the monitoring gather contextual information used by the analyzing part to decide whether an adaptation is needed or not; from this need, the planning part builds an execution plan to be executed by the executing part.

In figure 2, the grey boxes represent generic engines that can be implemented as interfaces, abstract classes or components; purple boxes represent specific implementation and white boxes are data structures. The notation is UML-like, meaning that arrows can be understood as "implements" and rhombus denotes composition. The dotted arrows represent data flow between the modules.

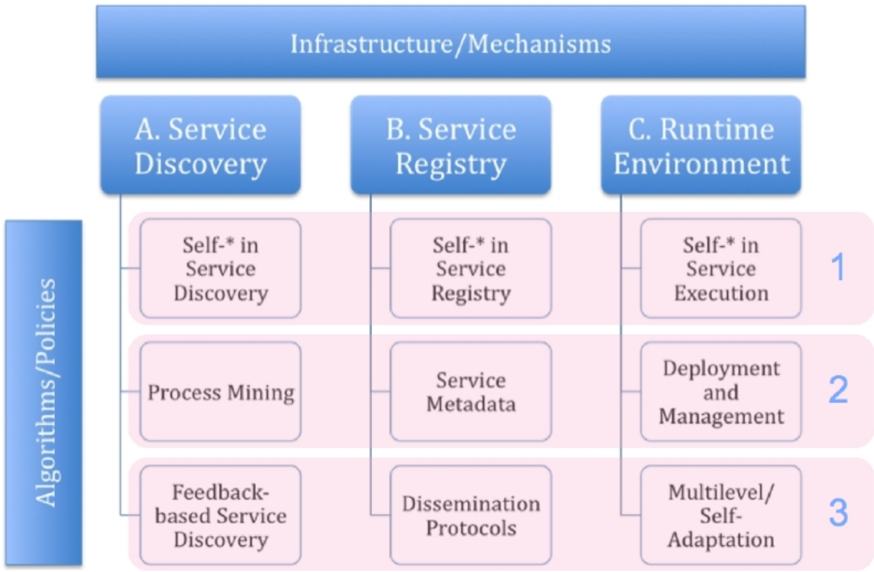

Figure 1: WP-JRA-2.3 Research Architecture

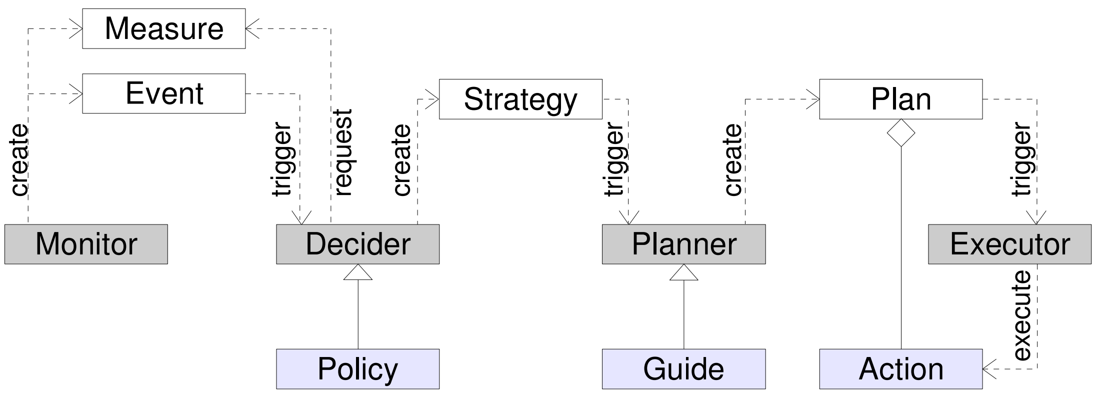

Figure 2: Structural decomposition of the general adaptation framework.

The contextual information is gathered by probes through *events* and *measures*. Events can trigger adaptation while measures are done on demand by the analyzing part when complementary information is needed. The monitoring is not only platform specific, it can also be application or domain specific when adaptation is not due to resources. The application itself can be monitored, for self-healing purpose for example, by a machine learning software, the user or by using ad-hoc metrics.

The analyzing part is done by the *decider*. When receiving an event, the decider chooses if an adaptation is needed by following a specific decision *policy*. This structure enables to choose the decider the best suited to each decision problem without imposing one way to write every algorithm. Furthermore, this structure enables to use the same decider for different services; only the policy is specific to each service adaptation.

Once an adaptation is chosen, the *decider* sends a *strategy* to the planning part, implemented by the *planner*. The planner has to work out how to apply the strategy to the service to adapt; the implementation of the planner refers to a *guide*. Which means that the decider has to decompose the received strategy into elementary tasks to be executed. In order to better know the current state of the service to adapt, the decider can request contextual measures to the monitoring part.

Then the *plan* of *actions* is sent to the executing part, which is the *executor*. Its role is to execute each action specified in the plan, taking into account the execution of the service to adapt. To do so, the executor may intercept the execution flow to execute adaptation actions.

The requirements set up for local adaptation and self-healing can be summarized as follows (for details see [11]).

Monitoring is responsible for providing information about at least three aspects of execution: (i) if the usage of resources compatible with the allocated resources; (ii) if the actual QoS of the service conform to the SLA and (iii) even if services are stateless from the client point of view, is this internal state healthy?

Decision support takes input from monitoring mechanisms and produces a strategy as a result of a decision. The Policy describes the way to implement decision mechanisms, it depends on the complexity and the accuracy of the decision to make and the response time of the system one wants to design. Such techniques range from simple Event-Condition-Action rule based system to complex learning algorithms.

Strategies must be able to differentiate and refine classes of actions to be triggered upon some events. *Adaptation* can alter some of the characteristics of the service. *Renegotiation* takes place if the service is unable to adapt its behavior or its parameters to maintain the required QoS therefore, (terms of the) QoS are renegotiated. In case no means of adaptation can be found and no negotiation is possible, the service could decide to *fail*.

Actions may realize various forms of adaptation. *Behavior adaptation* can change between different versions of a service if multiple instances exist, including a potential dynamic deployment. Another way to change a service's behavior is to use sub-services for each different behavior and to switch between them when appropriate. If one has access to different implementations of a service, many adaptations can be achieved inside the service container by changing the active implementation. *Parameter adaptation* is the easiest way for adapting a service, because the possible values and effects of a parameter should have been taken into account at the design time of the service. *Interface adaptation* is a way to support adaptation by dynamically changing interfaces but is not possible in all technologies.

3 Contributed papers

3.1 Overview

The core of this deliverable is constituted by a collection of published papers attached as appendices. These papers summarize research work carried out in or related to JRA-WP-2.3 according to its long-term goals. Albeit the deliverable is focused on decision support, the boundaries of the components of the conceptual framework are not crisp, see Section 2.3. Hence, with the goal of establishing decision support for local adaptation, research is also aimed at monitoring, strategy and action planning and other aspects. Roughly speaking decision making and decision support is embedded into the adaptation framework (Figure 2) with relations to nearly all aspects and requirements in Section 2.3 and [11]. The papers presented in this deliverable are not exclusively focused on 'decision support for local adaptation' but cover most of these facets while revealing different aspects of decision support related to local adaptation.

Table 1 summarizes the contributed papers and their position in the research framework. Research thread denotes the topic within the WP the given paper deals with. It can be a single topic (a box in Figure 1), a whole thread or the entire WP if there is no specific thread or topic it belongs to but addresses overall challenges and questions of the infrastructure research. Areas addressed refers to the definition and requirement analysis in [11] as summarized in Section 2.3.

Section 3.2 (Appendix A and Appendix B) drafts a conceptual architecture for an SLA-based virtual service provisioning. Due to its nature, adaptation (and partial self-* properties) may appear in many aspects at its three levels: negotiation, brokering and deployment. Decision making and decision support appears most evidently at the brokering level [22], [16]. This work is a direct continuation of [11], the conceptual architecture is built on the requirement analysis of the previous deliverable. Also, this work is a backbone of infrastructure research in WP-JRA-2.3.

Section 3.3 (Appendix C) targets a fundamental issue in decision support: monitoring that ensures the availability of actual and precise information. The proposed method focuses on QoS issues and is an integration of client-side and server-side approaches.

Section 3.4 (Appendix D) takes into consideration decision support by evaluating past experiences of a certain service. The paper is related to Section 3.3 yet, it is not a form of monitoring in a strict sense. One of the main contributions of the work is capturing non-trivial, non-functional features and parameters that must be part of QoS-aware service selection.

Section 3.5 (Appendix E and Appendix F) summarizes two papers related to monitoring and prediction as sources of information for decision support. [4] is a work that primarily related to research in WP-JRA-1.2 whereas [8] addresses challenges in WP-JRA-1.3. Both papers are strongly related to local adaptation and decision support for local adaptation by their monitoring, analyzing and prediction techniques. These two papers are also reported in a WP-JRA-1.2 and WP-JRA-1.3 deliverables, respectively.

Section 3.6 (Appendix G) introduces a framework for dynamic invocation of services so that service providers can be replaced at run-time allowing a transparent adaptation. Daios may serve as a general framework for various adaptation related experiments in S-Cube.

Section 3.7 (Appendix H) addresses local adaptation directly in a master-worker paradigm. The paper elaborates and presents a decision mechanism that is able to dynamically change the behavior of a service according to the environment. The decision is based on patterns (and thus, general) and involves learning abilities, i.e. the decision making mechanism is able to train itself for better tuned actions.

Section 3.8 (Appendix I and Appendix J) presents a higher level and more abstract mechanism to dynamic adaptation. A novel research thread is aimed at establishing nature metaphors for coordinating complex scenarios with many entities and uncertain or unknown conditions. Such an adaptation is hard (if not impossible) to formalize and steer by exact algorithms. On the other hand, nature analogons can give reasonable heuristics to reach states of equilibrium, minima or maxima according to the needs. The work presented in this section is a study on how service composition can be adapted by a chemical model. Both decision, strategies, policies and actions are encoded in the behavior of the chemical reactions that govern the state of the system. The work is preliminary with many plans for the future. Also, a former work is presented that is a study on dynamic workflow enactment (similar to service composition) in the chemical paradigm.

Table 2 summarizes the contributed papers by their scope of research and show which areas we found essential to advance local adaptation to more complex adaptation (for future deliverables.) On one hand there is an obvious need for data as an input of any decision support hence, a large bulk of investigation was devoted to various aspects of monitoring (analysis and prediction) partly as joint work with WP-JRA-1.2 (Appendix C, Appendix E and Appendix F). The other issue is what decision support can provide, what strategies or further actions (adaptation, adaptation patterns) can be derived from any decision making procedure (Appendix H). This latter problem is strongly related to the requirement analysis of self-healing infrastructures (see [11]) and – among other things –, a multi layered adaptation framework was drafted as an

Ref.	Paper	Research thread (Figure 1)	Partners	Areas addressed (Figure 2)
Section 3.2, Appendix A	A. Kertész, G. Kecskeméti, I. Brandic: An SLA-based resource virtualization approach for on-demand service provision. In Proceedings of the 3rd international Workshop on Virtualization Technologies in Distributed Computing (Barcelona, Spain, June 15 - 15, 2009). VTDC '09. ACM, New York, NY, 27-34. [22]	C, A1, B2, B3 of JRA-2.3	SZTAKI, TUW	monitoring, decision making, action planning
Section 3.2, Appendix B	A. Kertesz, J. D. Dombi, J. Dombi, Adaptive scheduling solution for grid meta-brokering, Acta Cybernetica Volume 19, pp. 105-123, 2009. [16]	JRA-2.3	SZTAKI, external	decision making
Section 3.3, Appendix C	A.Michlmayr, F.Rosenberg, P.Leitner and S.Dustdar: Comprehensive QoS Monitoring of Web Services and Event-Based SLA Violation Detection, to appear at 4th International Workshop on Middleware for Service Oriented Computing (MW4SOC), co-located with MIDDLEWARE'09, 2009 [7]	C2 of JRA-2.3	TUW	monitoring
Section 3.4, Appendix D	P.Leitner, A.Michlmayr, F.Rosenberg and S.Dustdar: Selecting Web Services Based on Past User Experiences, to appear at 2009 Asia-Pacific Services Computing Conference (APSCC 2009) [2]	A3, C2, C3 of JRA-2.3	TUW	decision, policy making
Section 3.5, Appendix E	B.Wetzstein, P.Leitner, F.Rosenberg, I.Brandic, F.Leymann, S.Dustdar: Monitoring and Analyzing Influential Factors of Business Process Performance. IEEE International Enterprise Computing Conference (EDOC), 2009 [4]	JRA-1.2	USTUTT, TUW	monitoring
Section 3.5, Appendix F	P.Leitner, B.Wetzstein, F.Rosenberg, A.Milchlmayr, S.Dustdar and F.Leymann: Runtime Prediction of Service Level Agreement Violations for Composite Services, to appear at 3rd Workshop on Non-Functional Properties and SLA Management in Service-Oriented Computing, co-located with ICSOC 2009 [8]	JRA-1.3	USTUTT, TUW	monitoring
Section 3.6, Appendix G	P. Leitner, F. Rosenberg and S. Dustdar. Daios: Efficient Dynamic Web Service Invocation. In IEEE Internet Computing, Vol. 13(3):72-80, 2009. [5]	JRA-2.3	TUW	adaptation in general
Section 3.7, Appendix H	F. André, G. Gauvrit, C. Pérez: Dynamic Adaptation of the Master-Worker Paradigm. International Conference on Computer and Information Technology [14]	C1 of JRA-2.3	INRIA	decision, strategy
Section 3.8, Appendix I	C. Di Napoli, M. Giordano, Zs. Németh, N. Tonellotto: A chemical metaphor to model service selection for composition of services. In Proceedings of the Second International Workshop on Parallel, Architectures and Bioinspired Algorithms (held in conjunction with PACT'09) [15]	C1 and C3 of JRA-2.3	CNR, SZTAKI	strategy, action
Section 3.8, Appendix J	M. Caeiro, Zs. Németh, T. Priol: Execution of Scientific Workflows based on an Advanced Chemical Coordination Model. Proceedings of PARA'08, Springer, to appear [18]	JRA-2.3	INRIA, SZTAKI, external	strategy planning

Table 1: Contributed papers

answer (Appendix A and Appendix B). The types of adaptation, the policies of adaptation, past experience add another facet of decision making (Appendix D, Appendix G). Finally, there is a need and a strong interest in finding a novel, high level model that captures more aspects of adaptation and self-* behavior (and thus, decision making is an integral part of it). Nature metaphors are good candidates to elaborate such models, Appendix I and Appendix J presents the first steps in this direction.

3.2 An SLA-based resource virtualization approach for on-demand service provision

In deliverable CD-JRA-2.3.2 [11] the focus was on research thread C "Runtime Environment" as shown in Figure 1. From the wide range of self-* properties, we detailed the requirements of self-healing and self-adaptation for a conceptual service architecture. Three closely related areas appear in this conceptual architecture, namely: agreement negotiation, service brokering and service deployment, and they are also closely connected to the other two research threads shown in Figure 1. In deliverable CD-JRA-2.3.2 [11] we examined this architecture and revealed the basic requirements for a future self-* realization of these core components. These basic requirements imply that there must be a negotiation phase when it is specified, what service is to be invoked and what are the conditions and constraints (temporal availability, reliability, performance, cost, etc.) of its use. Subsequently, an agent must select available resources that can be allocated for providing the services. These resources can be provided in many ways: clouds (virtualized resources configured for a certain specification and service level guarantees), clusters or local grids (distributed computing power with limited service level guarantees) or volunteer computing resources (no service level guarantees at all). Finally, the service must be deployed on the selected resources in an automatic manner.

In this deliverable we show, how these requirements can be fulfilled by proposing an SLA-based service virtualization (SRV) architecture [22] as shown in Figure 3. We target the same research questions as in deliverable CD-JRA-2.3.2 [11], and present a unified service virtualization environment called SRV that builds on three main areas: agreement negotiation, brokering and service deployment using virtualization. The relevant actors of the architecture are the following ones: A user, who wants to use a service, a Meta-Negotiator (denoted by MN in Figure 3), which is a component that manages Service-level agreements and mediates between the user and the lower level components by selecting the appropriate protocols for agreements, negotiating SLA creation, and handling fulfilment and violation. The Meta-Broker (MB) component is responsible for service mediation in terms of higher level brokering: its role is to select a broker (B) that is capable of deploying a service (S) with the specified user requirements. One level below, brokers interact with virtual or physical resources (R) and in case the required service needs to be deployed, they interact directly with the deployment service (Automatic Service Deployment), where deployment means installing the required service on the selected resource on demand. At the lowest level there are services that users want to deploy and/or execute, and resources (e.g. physical machines), on which virtual machines can be deployed/installed. The concepts of local adaptation summarized in Section 2.3, appear at all levels of the proposed service virtualization architecture. Therefore all components implement the four parts (namely the Monitor, Decider, Planner and Executor) of the general adaptation framework shown in Figure 2.

In this SRV approach users describe the requirements for an SLA negotiation on a high level using the concept of meta-negotiations. During the meta-negotiation only those services are selected, which understand specific SLA document language and negotiation strategy or provide a specific security infrastructure. This component uses monitoring during the execution of the negotiations, analysis for evaluating existing negotiation strategies, planning for selecting from

Problem addressed	Ref.	Paper
Decision and strategies: adaptation, adaptation negotiation, adaptation patterns	Section 3.2, Appendix A	A. Kertész, G. Kecskeméti, I. Brandic: An SLA-based resource virtualization approach for on-demand service provision. In Proceedings of the 3rd international Workshop on Virtualization Technologies in Distributed Computing (Barcelona, Spain, June 15 - 15, 2009). VTDC '09. ACM, New York, NY, 27-34. [22]
	Section 3.2, Appendix B	A. Kertesz, J. D. Dombi, J. Dombi, Adaptive scheduling solution for grid meta-brokering, Acta Cybernetica Volume 19, pp. 105-123, 2009. [16]
	Section 3.7, Appendix H	F. André, G. Gauvrit, C. Pérez: Dynamic Adaptation of the Master-Worker Paradigm. International Conference on Computer and Information Technology [14]
Interface adaptation, service selection (policies to decision support)	Section 3.4, Appendix D	P.Leitner, A.Michlmayr, F.Rosenberg and S.Dustdar: Selecting Web Services Based on Past User Experiences, to appear at 2009 Asia-Pacific Services Computing Conference (APSCC 2009) [2]
	Section 3.6, Appendix G	P. Leitner, F. Rosenberg and S. Dustdar. Daios: Efficient Dynamic Web Service Invocation. In IEEE Internet Computing, Vol. 13(3):72-80, 2009. [5]
Monitoring, prediction, analysis (input to decision support)	Section 3.3, Appendix C	A.Michlmayr, F.Rosenberg, P.Leitner and S.Dustdar: Comprehensive QoS Monitoring of Web Services and Event-Based SLA Violation Detection, to appear at 4th International Workshop on Middleware for Service Oriented Computing (MW4SOC), co-located with MIDDLEWARE'09, 2009 [7]
	Section 3.5, Appendix E	B.Wetzstein, P.Leitner, F.Rosenberg, I.Brandic, F.Leymann, S.Dustdar: Monitoring and Analyzing Influential Factors of Business Process Performance. IEEE International Enterprise Computing Conference (EDOC), 2009 [4]
	Section 3.5, Appendix F	P.Leitner, B.Wetzstein, F.Rosenberg, A.Milchlmayr, S.Dustdar and F.Leymann: Runtime Prediction of Service Level Agreement Violations for Composite Services, to appear at 3rd Workshop on Non-Functional Properties and SLA Management in Service-Oriented Computing, co-located with ICSOC 2009 [8]
High level models of dynamic adaptation	Section 3.8, Appendix I	C. Di Napoli, M. Giordano, Zs. Németh, N. Tonellotto: A chemical metaphor to model service selection for composition of services. In Proceedings of the Second International Workshop on Parallel, Architectures and Bioinspired Algorithms (held in conjunction with PACT'09) [15]
	Section 3.8, Appendix J	M. Caeiro, Zs. Németh, T. Priol: Execution of Scientific Workflows based on an Advanced Chemical Coordination Model. Proceedings of PARA'08, Springer, to appear [18]

Table 2: Contributed papers by addressed problem

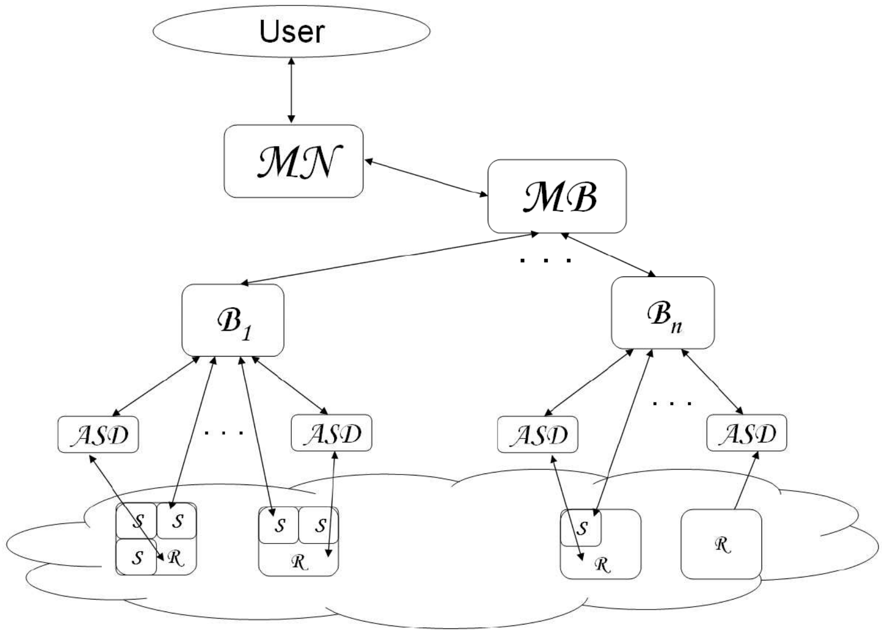

Figure 3: The SRV architecture.

the existing strategies and defining new ones, finally execution for utilizing the selected strategy. After the meta-negotiation process, a meta-broker selects a broker that is capable of deploying a service with the specified user requirements.

Scheduling at this level requires sophisticated approaches, because high uncertainty presents in resource and service availability. To ensure self-adaptability and enhance decision support at this level, we proposed an enhanced scheduling technique to improve broker selection in [16]. The Decision Maker component of the Meta-Broker is responsible for broker selection. It uses weighted functions with random number generation to select a good performing broker for user requests, even under high uncertainty. We have evaluated these algorithms in a simulation environment with real workload samples. The evaluation results presented in [16] show that the enhanced scheduling provided by the Decision Maker component enables better adaptation and results in a more efficient service execution.

The Meta-Broker also implements the Monitor component of the general adaptation framework shown in Figure 2 by tracking the states of the interconnected brokers. It also uses an analysis for comparing the performance results of the brokers. The Decision Maker subcomponent implements planning for determining a ranking of brokers according to the measured and analysed data, finally execution means invoking a broker with the highest rank. Once a broker is selected, it negotiates with virtual or physical resources with the help of the automatic service deployment (ASD) using the requested SLA document language and the specified negotiation strategy. Once the SLA negotiation is concluded, services can be deployed on the selected resource using the virtualization approach. The ASD uses monitoring for identifying defunct or overloaded service states, the analysis for making decisions about possible state transfers, planning for generating deployment descriptors, and finally execution for deployment, proxy installation and request forwarding. In [22] we have shown, how the target architecture can be realized, and demonstrated the utilization through a medical surgery scenario.

This work is closely related to the research topics on SLA-negotiation in WP-JRA-1.3 and on adaptation in WP-JRA-1.2. In the upcoming deliverable of WP-JRA-1.3 (CD-JRA-1.3.2) we further detail, how different policies can be defined and used for self-adaptation in the negotiation, brokering and deployment processes. In WP-JRA-2.3 we further continue and examine self-adaptivity and self-healing by specifying different strategies to improve adaptivity. These results will be included in a future deliverable CD-JRA-2.3.6.

3.3 Comprehensive QoS Monitoring of Web Services and Event-Based SLA Violation Detection

The foundation of both adaptation and decision support in service-based systems is the availability of accurate data, especially regarding Quality of Service (QoS) measurements. QoS is one of the main input of adaptive systems, which often use QoS as a trigger for adaptation or as some sort of target function (i.e., adaptation is done in order to improve QoS). Therefore, QoS needs to be continuously monitored. We have discussed this topic in [7].

Conceptually, there are two main approaches for QoS monitoring of Web services: Server-side monitoring is usually accurate but requires access to the actual service implementation, which is not always possible. In contrast, client-side monitoring is independent of the service implementation but the measured values might not always be up-to-date since client-side monitoring is usually done by sending probe requests (i.e., test requests that are similar to real requests). In [7] we aimed at combining the advantages of both approaches which has been realized in the Vienna Runtime Environment for Service-oriented Computing (VRESCo) [9]. Therefore, we have linked an existing client-side QoS monitoring approach [6] together with server-side monitoring based on Windows Performance Counters [10]. The main advantage of the client-side approach is that it is applicable for all services, and that its usage is not intrusive for the service. The disadvantage is that client-side monitoring as discussed in [6] introduces a notable overhead for the service, and the measurements may be somewhat inexact. The server-side approach, on the other hand, does not introduce a notable overhead and is very exact. However, its application is intrusive. Furthermore, since Windows Performance Counter technology has been used to implement the server-side monitoring approach in the paper it is only applicable to .NET-based Web services. In the paper, we aimed at integrating these two different types of QoS metrics under a coherent hood, in order to allow unified access to mesaurements produced using both approaches. This is done using an event-based approach. Finally, it is shown how this coherent QoS view can be used for SLA monitoring, however, not excluding other usage of the unified QoS data.

The place of this work in the overall vision of work package WP-2.3 is in the Deployment and Management research item. However, note that the essential outcome of the paper (accurate, integrated and up-to-date QoS information) is a precondition of many other approaches described in this deliverable. Additionally, QoS information is essential for all other research items in thread C of the WP-JRA-2.3 research vision. In the illustrative example in deliverable CD-JRA-2.3.3 [12] the case of a GPS unit is made. This unit utilizes Web services to retrieve essential information, which it needs to function, e.g., up-to-date traffic information. This information needs to be available in near-realtime (nobody wants to learn about a traffic jam when already stuck in it), therefore, the QoS of the traffic services is important for the functioning of the GPS unit. Using the means discussed in the paper the unit can monitor the QoS of the traffic services, and switch to a better performing one if the quality of the used one degrades. In terms of Figure 2 this paper implements the *Monitor* component, which emits (QoS) measurements. The VRESCo event engine can also be used to emit events, as depicted in Figure 2.

This paper can also be seen as a major integration point with other work packages within the S-Cube project: the work conducted in WP-JRA-1.3 (quality provisioning and SLA conformance) very much depends on accurate underlying QoS information, and adaptable service compositions as discussed in WP-JRA-2.2 also need quality metrics to inform decision components and trigger adaptation. In that sense this paper fulfills the promise of WP-JRA-2.3 that the work package will provide fundamental services that other work packages need for their (higher-level) functions. The work described here also has some dependencies on WP-JRA-1.2. However, while WP-JRA-1.2 focuses on principles and methodologies for monitoring we focus more on the concrete implementation of monitoring, and the research issues arising from the practical implementation of QoS monitoring.

3.4 Selecting Web Services Based on Past User Experiences

Adaptation in service-based systems often means exchanging one service for another (defined as local adaptation within the scope of work package WP-JRA-2.3), or by exchanging some parts of a service composition with another service or composition (multi-level adaptation). For this to be possible advanced service selection mechanisms need to be available, which take the quality of services (QoS) into account. This is related to the contribution of [7], which has been discussed before. We have discussed selection of services in a large-scale service environment (the Internet of Services) in [2].

In [2], we looked at the problem of QoS-aware service selection from a different angle. In this work we extended the notion of QoS to include not only technical parameters of the service execution (response time, availability, etc.) but also various other (harder to capture) factors which influence the service quality as experienced by the service end user (Quality of Experience, QoE). The problem of QoE is that it is inherently hard to quantify and measure automatically. However, the Web 2.0 trend of folksonomies provides a reasonable way to capture QoE by incorporating human factors, and evaluate the quality of services based on feedback from earlier transactions, e.g., using numerical ratings. Additionally, folksonomies often make use of the concept of emergent semantics (i.e., determining the semantics of resources through large amounts of independent, publicly available metadata, such as tags) to functionally describe resources. The main contribution in our work is the combination of these two concepts into a service selection framework, which captures human-perceived QoE using both free form tags (unstructured feedback similar to Web 2.0 tags) and numerical ratings (structured feedback, i.e., ratings) to narrow down and rank available service alternatives. In the paper we tackle practical challenges such as tag merging, preventing spamming of service providers or including the context of invocations. Furthermore, we explain a system implementation based on the VRESCo [9] SOA runtime which realizes these ideas.

This work is an important first step in the Feedback-based Service Discovery research item in the work package's research vision. In essence, the work described in the paper implements a recommender system for Web services, therefore, we argue that our results are clearly related to decision support with the aim of adapting service-based systems. This work also integrates well with the illustrative example in deliverable CD-JRA-2.3.3. In this example the GPS unit is expected to integrate information services from the 'Internet of Services', which obviously asks for good means to actually select them (or to decide between a number of similar services). This use case is very similar to the one used in the paper (travel planning). In fact, the travel planning use case in [2] is a direct offspring of the illustrative example in CD-JRA-2.3.3 [12]. This paper is related to the general adaptation framework in Figure 2 in that the service selection mechanisms implement the *Guide* component, which supports the *Planner* in finding optimal adaptation actions.

Just like [7] described before, this paper implements some promises of the work package: namely, that WP-JRA-2.3 will provide advanced service selection mechanisms on which other work packages can build.

3.5 Monitoring and Analyzing Influential Factors of Business Process Performance and Runtime Prediction of Service Level Agreement Violations for Composite Services

At its core, decision support for local adaptation is about information, i.e., (1) information about potential or current problems (e.g., SLA or KPI violations in a business process) and (2) information about possible improvements. S-Cube members of Vienna University of Technology (TUW), University of Stuttgart (USTUTT), and the Commonwealth Scientific and Industrial Research Organization (CSIRO, Australia) have recently worked on this information problem, and produced two relevant papers [4, 8], which address the first information problem discussed above.

The first problem has been addressed in [4] (see deliverable CD-JRA-2.2.2 for details). In this paper we have presented an integrated framework for run-time monitoring and analysis of the performance of WS-BPEL processes. The main contribution of the paper is a framework for dependency analysis, i.e., a machine learning based analysis of PPMs (metrics based on process runtime data, e.g., "number of orders which can be served from inhouse stock") and QoS metrics, with the ultimate goal of discovering the main factors of influence of process performance (KPI adherence in the case of this paper, but a generalization to other targets is possible). These factors are visualized in an easy-to-interpret decision tree, which we refer to as the dependency tree). We present the general concepts of our analysis framework, and provide experimental results based on a purchase order scenario, identify cases when dependency trees do not show expected results, and explain strategies how these problems can be coped with.

Dependency trees help a human business analyst understanding why business processes (or service compositions) do not perform as well as expected. However, this analysis is exclusively post mortem, i.e., problems can only be detected after they have started to (significantly) hurt the business performance. In that sense an ex ante analysis would be preferable, which would help predicting performance problems while it is still possible to undertake actions to either alleviate or prevent them. In [8] we have extended our results from [4] to predict SLA violations at runtime. This is done by analyzing both QoS and process instance data (PPMs as described above), and using estimates to approximate not yet available data. The ideas presented there are most applicable for long-running processes, where human intervention into problematic instances is possible. Our system introduces the notions of checkpoints (points in the execution of the composition where prediction can be done), facts (data which is already known in a checkpoint, such as the response times of already used services) and estimates (data which is not yet available, but can be estimated). Just like in [4], we use machine learning techniques (in this case neural networks, i.e., multilayer perceptrons) to implement the actual analysis. This work can be seen as a first step - currently, the actual adaptation has to happen manually. In the future we plan to extend these works to adapt business processes and composite Web services in an automated way.

These works are orthogonal to the work packages research agenda, since they do not fit directly into one of the research items in the WP-JRA-2.3 research vision. However, the results presented here are directly related to (or even a precondition for) research in the items Deployment and Management, and Multilevel/Self-Adaptation. Additionally, our chosen machine learning approach has some relevance for the research item Process Mining, which will use similar techniques to achieve its goals. With regard to the adaptation framework in Figure 2 these works feed into both the *Decider* and *Planner* components.

These works also form a major integration point for the work package with the rest of the S-Cube project. The ideas presented here also fit the scope of WP-JRA-1.2, which deals with monitoring methodologies for service-based applications. Additionally, the outcomes of this work are important for WP-JRA-1.3, since our results ultimately help achieving SLA compliance. On a related note the results of this work can be used by WP-JRA-2.1 to enable management and performance monitoring of business processes. Finally, the analysis results generated in this paper are a cornerstone of adaptable service compositions as discussed in WP-JRA-2.2.

3.6 Daios: Efficient Dynamic Web Service Invocations

In S-Cube, we define local adaptation of service-based systems as the process of exchanging one service binding for another. However, this is not easy to do with current state of the art Web service tooling, e.g., Apache CXF [3]. Today, services are generally used by means of precompiled stubs, which are deeply integrated with the client application code. For local adaptation, a less tightly coupled approach is necessary. We have covered this problem in [5].

We have presented an approach, which aimed at resolving this fundamental issue in [5]. There, we presented Daios, a dynamic service invocation framework that enables application developers to create stubless and dynamic service clients, which are not strongly coupled to a specific service provider. Instead, the Daios dynamic interface offers a higher degree of provider transparency that allows service providers to be exchanged at run-time. We explain how the requirements of stubless service invocation, protocol independence, message orientation, client-side asynchrony and performance have driven the development of Daios, and what the basic architecture of the system looks like. Most importantly, we present some details about Daios' matching of so-called DaiosMessages (a high-level data encoding format proprietary to Daios) to concrete SOAP or RESTful Web services. Finally, we give usage examples of the Daios framework. Daios is available both for Java and the .NET platform. In the meantime, we have integrated the Daios Web service stack with our larger Web service infrastructure VRESCo [9], where Daios is now exclusively used as a powerful service invoker.

The capabilities provided by Daios are also much needed in the work package's illustrative example as discussed in CD-JRA-2.3.2 [11]: in this scenario the usage of 'ad hoc services' to provide added-value features (such as e.g., an on-demand streaming service) is described. Such ad hoc services cannot be consumed easily without means to dynamically invoke services which are unknown at design-time. Daios allows to use such services, and abstracts away from implementation details of ad hoc services (such as e.g., whether the service is SOAP-based or uses the REST architectural style). In terms of Figure 2 Daios is an essential part of the *Executor* component, since the execution of rebinding actions rely on the availability of means for dynamic service invocations.

In terms of integration with other work packages in S-Cube, this paper is important in that it provides basic infrastructure service, which other work packages or partners can build upon. This is specifically important for WP-JRA-2.2, which has to deal with adaptable service compositions. Without means to provide a loose coupling of service clients and service providers this adaptivity is not achievable.

3.7 Dynamic Adaptation of the Master-Worker Paradigm

The paper "Dynamic Adaptation of the Master-Worker Paradigm" [14] main contribution is the presentation of a decision algorithm to dynamically adapt the Master-Worker paradigm. This paper focuses on Thread C (service runtime) and Challenge C1 (Self-*) of the WP JRA-2.3 research framework.

The load of a (web) service can be very dynamic and unpredictable, especially when services are dynamically discovered, so a service may become part of many different service compositions at the same time. In this case a local adaptation of such a service is needed in order for the service to be able to still meet its SLA and/or to negotiate new SLAs. This has a deep impact on the implementation of services : in order to be able to serve more requests a service should be able to use more computing power when needed. This is only possible if using a parallel computing platform such as a cluster or a Grid.

This paper emphases the local adaptation of one service at the infrastrucutre level. It is one of the basic used at the infrastructure level for grid-based assembly simulation in manufacturing process (see section 3.10 in PO-IA-3.2.1)

However, the size, heterogeneity and dynamism of these execution platforms make there use quite complex. Furthermore, today there is no effective and relatively simple solution to programming applications independently of the target architectures. In any case, one should use some well known parallel programming paradigm. Among them, the master-worker proposes provides a high level abstraction of those platforms for distributing the computational load among a set of "workers" processes in a portable way. This paradigm is well known and has been used for years and has different alternative implementations depending on the way to distribute the work and on the communication between the master and the workers.

In this paper, we propose to make the master-worker abstraction dynamically adaptable. After having described the main alternatives that can be used to implement the master-worker paradigm and shown that each is better fitted to different situations, our objective is to dynamically change the implementation pattern at run time, depending on the current state of the execution environment. More specifically, this paper first characterizes the master-worker paradigm on distributed platforms, by specifying different parameters. Then the decision algorithm, based on the description of the behavior of the patterns, which depends on the state of the pattern and of the distributed system, and on a QoS objective, is presented. This algorithm is designed to be generic: new patterns can be added to the application without modifying the existing parts of the algorithm implementation. It may also be added to a learning mechanism which could tune its decisions according to the results of preceding simulations that has been used to validate it are briefly presented.

3.8 A chemical metaphor to model service selection for composition of services

In service-oriented systems more service providers can provide the same functionality at different conditions, or even the same provider can provide one functionality at different conditions. These conditions refer to non-functional characteristics of a service (like price, time to deliver, and so on) that can change in time depending on both provider policies and consumer's needs. For example, the cost associated to a service may vary according to market conditions, or the time to deliver may vary according to the workload of the provider. Given the dynamic nature of service non-functional characteristics, it is necessary to provide a run-time infrastructure able to adapt autonomously and dynamically to changing conditions.

The paper "A chemical metaphor to model service selection for composition of services" [15] investigates the possibility of using the chemical metaphor [21] to provide service selection mechanisms that adapt in response to dynamic requirements and circumstances for SBAs, and as such it addresses the Runtime Environment Research Thread (C) and in particular the Challenge Self-* in Service Execution (C1) of WP-JRA-2.3. A forerunning study in the same field but focusing on dynamic workflow modeling can be found in [18].

Nature metaphors may suggest computational models to develop new runtime mechanisms for SBAs in order to support local adaptation. By imitating nature processes, certain parameters are governed by nature laws that are known to evolve towards equilibrium or energy minimum, and living entities may cooperate, share information, reproduce or die, and so on. Among these metaphors, the chemical is interesting since matter transforms only by local interactions. Each such interaction (their precise location, time and order) is a primitive and unpredictable step but their overall effect governs the matter into a predictable state. In the chemical model of computation, chemical reactions involve molecules (i) that are within each others proximity (ii) and their actual conditions (chemical properties, temperature, energy, etc.) enable the reaction.

We propose to model service instances together with non-functional characteristics (i.e. the temporal constraints on its delivery) as service offers (bids) represented by chemical primitive molecules, and the rules to combine these instances as chemical reactions. This approach allows to capture service selection in the context of SBAs in terms of local rules. Once a preliminary aggregation of services a SBA is computed (i.e. the chemical system reaches an inert state), it is possible to bootstrap a new selection process once new offers are available or already existing offers are changed (e.g. because the temporal constraints of a service changed). These new offers are new molecules of the chemical system that may trigger new reactions. The new molecules, by following the same local rules, may be selected to form a new aggregation of services with existing molecules, so that the system exhibit an adaptive behavior that is not based on centralized and/or coordinated policies. Moreover, the local behavior of molecules together with the overall constraints of the aggregation may produce adaptation of the system to different configurations that are not planned in advance. In this way service selection is modeled as an evolving process where the evolution consists of including the possibility to refine the selection according to new offers that become available either because new service instances become available, or existing service instances change the parameter value of their offers.

The presented approach addresses local adaptability of service selection in the context of aggregation of services that compose an SBA, so the work is related also to the challenge of WP-JRA-2.2. Furthermore the approach refers to non-functional characteristics of services that may change dynamically (such as the temporal availability) so that connections with WP-JRA-1.2 and WP-JRA-1.3 can be foreseen. The possibility to select different services with different non-functional characteristics that better satisfy the requirements of a composition of services, relates to the research challenge "Multilevel and Self-adaptation" in the WP-JRA-2.3 vision.

4 Conclusions

This deliverable summarizes the progress of research in WP-JRA-2.3 related to self-* and adaptive infrastructure, with a special emphasis on decision support for local adaptation. This deliverable is partly based on the outcome of CD-JRA-2.3.2 "Basic requirements for self-healing services and decision support for local adaptation" [11] and is a prerequisite for CD-JRA-2.3.6 "Specifications of policies and strategies for local adaptation" as well as CD-JRA-2.3.8 "Specifications of policies and strategies for distributed and multi-level adaptation". Due to the nature of the topics this deliverable addresses, there are strong ties to WP-JRA-1.2 (Monitoring and Adaptation) and WP-JRA-1.3 (Quality Provision and SLA Conformance); some papers are related to more than one WPs and deliverables.

The deliverable is a collection of scientific papers published in conference proceedings and journals unified along the research directions of WP-JRA-2.3. They all underwent a review that ensures their technical soundness and novelty thus, they demonstrate progress in the WP. Since decision support is just one aspect of the whole adaptation/self-*, papers address a much wider scope within the adaptation framework (Figure2) while decision, decision support and the related fields are parts of the work. The positioning of the papers within the adaptation framework, their relationship to the WP-JRA-2.3 research goals and vision and to other research WPs are exposed in Section 3. Section 3 also reveals that a number of research threads and challenges in the research architecture (Figure 1) have been addressed as well as most of the issues in the requirement analysis [11] and components of the adaptation framework (Figure 2) are targeted at.

References

[1] Integration Framework Baseline, S-Cube Deliverable CD-IA-3.1.1, 2009

[2] P.Leitner, A.Michlmayr, F.Rosenberg and S.Dustdar: Selecting Web Services Based on Past User Experiences, to appear at 2009 Asia-Pacific Services Computing Conference (APSCC 2009), 2009

[3] http://cxf.apache.org/

[4] B.Wetzstein, P.Leitner, F.Rosenberg, I.Brandic, F.Leymann, S.Dustdar: Monitoring and Analyzing Influential Factors of Business Process Performance, presented at IEEE International Enterprise Computing Conference (EDOC), 2009

[5] P.Leitner, F.Rosenberg, S.Dustdar, Daios: Efficient Dynamic Web Service Invocation, IEEE Internet Computing, vol. 13, no. 3, pp. 72-80, May/June 2009, doi:10.1109/MIC.2009.57

[6] F. Rosenberg, C. Platzer, S. Dustdar: Bootstrapping Performance and Dependability Attributes of Web Services. In: Proceedings of the IEEE International Conference on Web Services (ICWS'06), 18-22. September 2006, Chicago, USA.

[7] A.Michlmayr, F.Rosenberg, P.Leitner and S.Dustdar: Comprehensive QoS Monitoring of Web Services and Event-Based SLA Violation Detection, to appear at 4th International Workshop on Middleware for Service Oriented Computing (MW4SOC), co-located with MIDDLEWARE'09, 2009

[8] P.Leitner, B.Wetzstein, F.Rosenberg, A.Michlmayr, S.Dustdar and F.Leymann: Runtime Prediction of Service Level Agreement Violations for Composite Services, to appear at 3rd Workshop on Non-Functional Properties and SLA Management in Service-Oriented Computing, co-located with ICSOC 2009, 2009

[9] A. Milchmayr, F. Rosenberg, P. Leitner, S. Dustdar: End-to-End Support for QoS-Aware Service Selection, Invocation and Mediation in VRESCo. Technical Report (TUV-1841-2009-03), Technical University Vienna. Currently under review at IEEE Transactions on Services Computing

[10] http://msdn.microsoft.com/en-us/library/aa373083(VS.85).aspx

[11] Requirement Analysis for Decision Support and Local Adaptation. S-Cube deliverable CD-JRA-2.3.2, March 2009.

[12] Requirements for service registries in dynamic environments and evaluation of existing service registries, S-Cube deliverable CD-JRA-2.3.3, February 2009

[13] State of the art report, gap analysis of knowledge on principles, techniques and methodologies for monitoring and adaptation of SBAs, S-Cube deliverable PO-JRA-1.2.1, July 2008

[14] F. André, G. Gauvrit, C. Pérez: Dynamic Adaptation of the Master-Worker Paradigm. International Conference on Computer and Information Technology, IEEE, pp. 185-190.

[15] C. Di Napoli, M. Giordano, Zs. Nmeth, N. Tonellotto: A chemical metaphor to model service selection for composition of services. Proceedings of the Second International Workshop on Parallel, Architectures and Bioinspired Algorithms (held in conjunction with PACT'09), pp. 11-19.

[16] A. Kertész, J. D. Dombi, J. Dombi, Adaptive scheduling solution for grid meta-brokering, Acta Cybernetica Volume 19, pp. 105-123, 2009.

[17] J. O. Kephart, D. M. Chess: The vision of autonomic computing. IEEE Computer Vol. 36 No. 1, 2003, pp. 41-50.

[18] M. Caeiro, Zs. Németh, T. Priol: Execution of Scientific Workflows based on an Advanced Chemical Coordination Model. Proceedings of PARA'08, Springer, to appear

[19] J. Buisson, F. André, J-L. Pazat: Dynamic adaptation for Grid computing. P.M.A. Sloot, A. Hoekstra, T. Priol, A. Reinefeld, M. Bubak (eds.) Advances in Grid Computing - EGC 2005 (European Grid Conference, Amsterdam, The Netherlands, February 14-16, 2005, Springer

[20] Y. Becerra, D. Carrera, E. Ayguado: Batch Job Profiling and Adaptive Profile Enforcement for Virtualized Environments, Proceedings of 17th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP 2009), Weimar, Germany

[21] J.-P. Banâtre, P. Fradet, D. Le Metayer: Gamma and the chemical reaction model: Fifteen years after. In: Multiset Processing, Lecture Notes in Computer Science 2235, Springer 2001

[22] A. Kertész, G. Kecskeméti, I. Brandic: An SLA-based resource virtualization approach for on-demand service provision. In Proceedings of the 3rd international Workshop on Virtualization Technologies in Distributed Computing (Barcelona, Spain, June 15 - 15, 2009). VTDC '09. ACM, New York, NY, 27-34.