Grant Agreement N° 215483

Title:	*CD-JRA-2.3.6: Specifications of policies and strategies for adaptation*
Authors:	*CNR, INRIA, SZTAKI, TUW*
Editor:	*Zoltan Juhasz (SZTAKI), Zsolt Nemeth (SZTAKI)*
Reviewers:	*Jean-Louis Pazat (INRIA)* *Fabrizio Silvestri (CNR)* *Branimir Wetzstein (U-STUTT)*
Identifier:	*CD-JRA-2.3.6*
Type:	*Deliverable*
Version:	*1.0*
Date:	*February 25, 2011*
Status:	*Final*
Class:	*Internal*

Management summary

This deliverable aims to present the research progress of the project partners since the establishment of the decision support techniques and methodologies for local adaptation in deliverable CD-JRA-2.3.4. This progress was focusing on the various aspects on the infrastructure that supports the creation of policies and strategies for local adaptation in service-based applications. The research results are presented through the summaries of joint papers that are classified according to the Integrated Research Framework of the project.

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).

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 todays 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/`

Foreword

This deliverable, "*CD-JRA-2.3.6: Specifications of policies and strategies for adaptation*" aims to address the following goals:

– to continue, refine and consolidate the initial results of CD-JRA-2.3.4 towards the initial infrastructure level techniques that support the specification of policies and strategies for adaptation.

– to present the integration results through the summaries of the joint papers that target the different aspects of the infrastructure level adaptation problem.

– to lay down the cornerstones of the future JRA-2.3 deliverables that would extend the introduced techniques to support distributed and multi-level adaptation when applied to distributed systems like grid or cloud infrastructures.

Chapter 1

Introduction

This document presents deliverable CD-JRA-2.3.6: "Specifications of policies and strategies for adaptation" of the Work Package "WP-JRA-2.3: Research on Service Infrastructures" of the S-CUBE project. Task 1 in this WP explores infrastructure mechanisms for the run-time adaptation of services, hence the work reported in the current deliverable continues to deal with open problems in this area in order to further improve our understanding of the behaviour of complex service systems as well as to develop methodologies for the systematic design and implementation of adaptable and self-healing service-based applications. As a result of earlier research work, we discovered that we should also consider models from non conventional programming methods, such as chemical programming for adaptable service infrastructures. Thus, significant effort was devoted to studying the applicability of chemical models for some aspects of adaptation; a major part of the published papers and this deliverable addresses this issue and presents the preliminary results.

The deliverable builds on the results and continues the work reported in the two previous deliverables of this work package. CD-JRA-2.3.2 "Basic requirements for self-healing services and decision support for local adaptation" captured the requirements for the design and realization of self-healing services within the S-Cube architecture while deliverable CD-JRA-2.3.4 "Decision support for local adaptation" documented research results related to making services adaptable and self-aware while adhering to a set of given specifications and mechanisms; it also investigated the applicability of certain policies to trigger local adaptation mechanisms.

The scope of the research reported here is policies and strategies for adaptation from the infrastructure point of view. The focus is not on specification of certain policies and strategies, but rather on the elements of infrastructure mechanisms that enable or support the specification of such policies and strategies and their application. In this aspect, the work presented in this deliverable is a pre-requisite for specifying policies and strategies and thus, strongly connected to research in other work packages like WP-JRA-1.2 and WP-JRA-2.2 (as the distribution of papers in Table 2.2 proves); in some sense, it opens the possibility to apply certain methods developed in those WPs.

This deliverable is a bridge towards CD-JRA-2.3.8 "Specifications of policies and strategies for distributed and multi-level adaptation" and to a lesser extend to CD-JRA-2.3.9 "QoS and SLA aware service runtime environment" as well. As it was the case in previous deliverables, it presents the current stage of research that started from requirements of simple local adaptation to issues of multi-level adaptation and as such, the work is an ongoing process where boundaries of the current stage are blurred. The papers presented in the deliverable address two of the challenges in WP-JRA-2.3, "Multi-level and self-adaptation" and "Deployment & Execution Management" involving various elements of the IRF (see Tables 2.1 and 2.2). The deliverable is strongly connected to other WPs, namely WP-JRA-2.2, addressing "QoS-aware Adaptation of Service Composition", "Formal Models for QoS-Aware Service Compositions"; WP-JRA-1.3

addressing challenge "Proactive SLA negotiation and agreement" and WP-JRA-1.2 addressing "Comprehensive and integrated adaptation and monitoring principles, techniques, and methodologies". The connection in the latter two cases is stronger; there is a natural relation and co-operation between WP-JRA-2.3 and WP-JRA-1.2 and also, some of the papers (or the ideas presented in the papers) are eligible for deliverables in WP-JRA-1.2 and WP-JRA-1.3. A detailed analyisis of papers and their relationship to the IRF is presented in Chapter 2.

The research work and the bulk of this deliverable is reported in the form of joint publications of researchers within the S-CUBE project. The next sections of the introduction provide a short overview of the S-CUBE Work package JRA-2.3 Research Architecture and discuss the role of policies and strategies in adaptation. Chapter 2 contains a summary of each contributed paper highlighting the main contributions and its relevance to the research topic. The full papers can be found in the Appendix. The document ends with the conclusions in which we discuss the results and outline the next steps of our research.

1.1 The JRA-WP-2.3 research architecture

Research work in WP-JRA-2.3 is driven by the Work Package vision that summarizes the research challenges and structures the research work. Figure 1.1 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.

The work carried out in the current period and reported in this deliverable mostly concerns research runtime issues and hence, is related to research thread C.

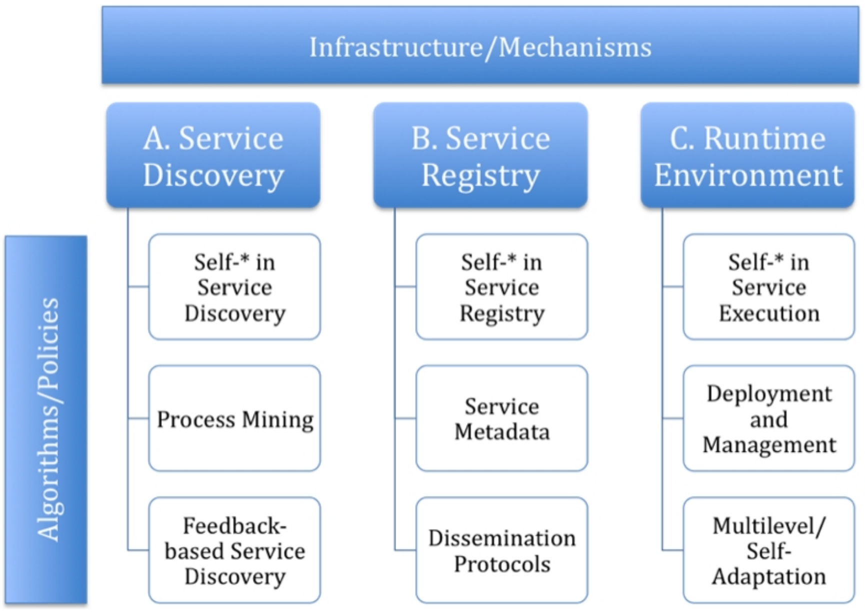
Infrastructure/Mechanisms
Algorithms/Policies
A. Service Discovery
B. Service Registry
C. Runtime Environment
Self-* in Service Discovery
Process Mining
Feedback-based Service Discovery
Self-* in Service Registry
Service Metadata
Dissemination Protocols
Self-* in Service Execution
Deployment and Management
Multilevel/ Self-Adaptation

Figure 1.1: WP-JRA-2.3 Research Architecture

1.2 Concepts of policies and strategies for 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 in the deliverable JRA-2.3.2 "Basic requirements for self-healing services and decision support for local adaptation" [1].

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

Decision support takes input from monitoring mechanisms and produces a strategy as a result of a decision. There are many ways to implement decision mechanisms depending 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 when the service is unable to adapt its behavior or its parameters so as to maintain the required QoS, therefore, terms of the QoS should be 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 it is not possible in all technologies.

The adaptation framework (Figure 1.2) follows the functional decomposition of an autonomic manager suggested in [4], classifying the adaptation in *monitoring*, *analyzing*, *planning* and *executing* parts. Each part triggers the next one: monitoring gathers contextual information used by the analyzing module to decide whether adaptation is required or not; from this need, the planner creates an execution plan that will be carried out by the executor.

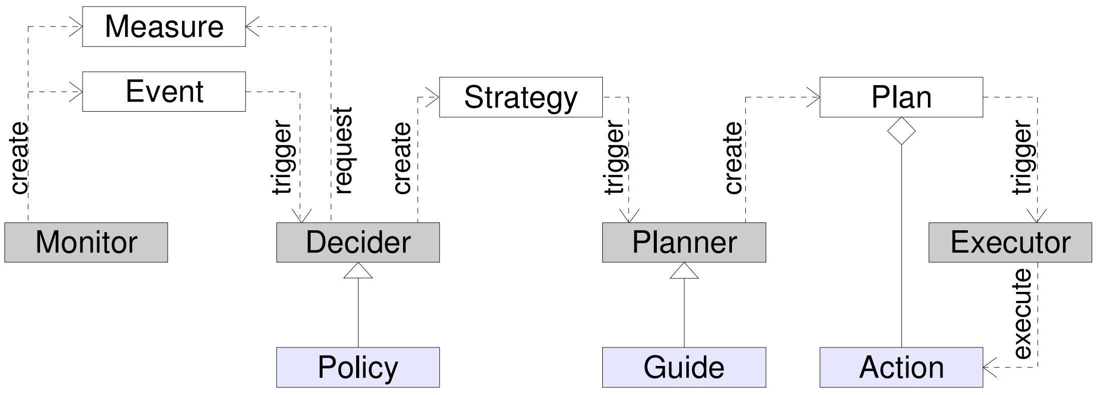

Figure 1.2: Structural decomposition of the general adaptation framework.

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. 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 also be monitored, for example for self-healing purposes, by a machine learning module, 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 the decider to choose the best suited method to each decision problem without imposing one way to write every algorithm. Furthermore, this structure allows one to use the same decider for different services; only the policy needs to be 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. This means that the planner has to decompose the received strategy into elementary tasks for execution. 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 action plan is sent to the executing part, to the *executor*. The role of the *executor* 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 current focus of our research reported in this deliverable is on issues related to the question of what elements are necessary to enable the specification and enactment of policies, and how to use adaptation strategies in service runtime systems. Adaptation at the service infrastructure level has to be supported by various means. Resource allocation and load balancing, scheduling are pre-requisites for adaptive execution and enforcement of execution policies. Health management is responsible for service node monitoring and - if required - triggering actions for restarts, replication, migration, and so on, in order to maintain service availability. Most of these concepts are well-known and in use in distributed systems. New problems arise, however, in the presence of virtualisation, service-level agreements, specification of execution and adaptation policies, for example scheduling service executions with licensing, on-demand service-provisioning under QoS constraints, policy based resources usage and cost control.

Large-scale service systems may use other forms of policy and strategy specification and enforcement mechanisms. Non-conventional paradigms, such as the chemical metaphor allows us to specify some elements of adaptation strategies and policies implicitly.

Chapter 2

Contributed papers

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, mainly driven by research plans in Task 1 ”Infrastructure Mechanisms for the Run-Time Adaptation of Services”. First, it is briefly sketched how these papers are related to elements and aspects of establishing policies and strategies at infrastructure level, what mechanisms they provide for specifying and enacting them and how they are related to general adaptation requirements. It is important to notice that the work is not aimed at specifying policies and strategies rather, at the elements and operations of the infrastructure that enable their specification. As usual, the boundaries of the research work are not strict, policies and strategies cannot exist on their own but as part of a larger adaptation framework (Figure 1.2) hence, the presented work covers nearly all aspects of adaptation as explained in Chapter 1.2 and [1].

Table 2.1 summarizes the contributed papers and positions them using the Integrated Research Framework (IRF) [3] based on the IRF elements they are related to. Most papers belong to research thread C (Figure 1.1) hence, this category is not shown in the table. Table 2.2 is an orthogonal table: it shows what challenges of the IRF are addressed and how the papers are distributed among IRF elements within a challenge. This table is a ”map” of papers showing how they cover the research goals of WP-JRA-2.3, how they are related to each other (if they are close or distant in terms of the IRF) and if they provide a broader coverage of some problems or they focus on some particular items. The table also shows how papers (and research work) are related to other WPs: it can be clearly seen that challenges in WP-JRA-1.2, WP-JRA-1.3 and WP-JRA-2.2 are also addressed.

The work presented in [15] (Chapter 2.1) proposes an approach called Service-level agreement-based Service Virtualization (SSV) that addresses autonomic service execution in heterogeneous, distributed service-based environments. It also discusses how the principles of autonomic computing are incorporated to the SSV architecture to cope with the error-prone virtualization environments. The autonomic behaviour of SSV is validated in a Cloud simulation environment, using a biochemical application as a case study.

Paper [11] (presented in Chapter 2.2) provides explicit examples on implementing strategies and policies for software license allocations. Licenses are handled similarly to traditional hardware resources during resource allocation, and thus they can be reserved for selected job executions. The scheduling of licenses is achieved via multi-agent negotiation techniques, allowing the combination of local reasoning results from various viewpoints into a single scheduling decision. Service providers may inject custom priorities at several points into the scheduling process including license selection (which licenses are consumed first), date and host selection in the local environment, aggregation of possible schedules (e.g. which resource provider is preferred). All these are achieved via logical rules working on the semantic resource reservation data.

The work in [14] (Chapter 2.4) proposes a QoS assurance framework that facilitates QoS management for services built on distributed infrastructures, such as grids and clouds. The framework provides a rich set of QoS management functions and supports dynamic adaptation of service behavior and resource usage in order to maintain agreed service levels. It demonstrates the interaction of the monitoring, analysis, planning and execution modules – as described in our reference architecture in Figure 1.2 – in providing lossless audio to service customers by means of adaptation.

The work presented in [9] (Chapter 2.3) proposes an application level scheduling module based on the use of *continuations* [16]. It allows providers to adapt service execution at the application level according to requirements that come from both users and providers strategies. In this work, users specify their requirements in terms of the "cost" they are willing to pay to obtain the service results, while providers execute the required services according to a policy that takes into account the user request and the workload in terms of number of service execution requests it receives. In addition, services can be deployed with reconfigurable features by invoking scheduling module interfaces that support service execution suspension and resuming on demand, so making it possible to change the resources allocated to the service during its execution. The interfaces may be used by other components of the S-Cube infrastructure to implement strategies and policies specified at the other levels.

The work in [7] (Chapter 2.5) is a sort of a study for modeling workflows (and hence, service compositions) with strong emphasis on dynamicity. This work is an additional support yet, orthogonal in some sense to 2.6 and [6]. The aim of the work is to conceptualise a workflow execution engine on chemical modeling principles that allows all sorts of dynamic changes in workflow structure, instance, resource mapping and advance workflow constructs that are not precisely known at design time. These mechanisms may enable the enactemnt of various adaptation strategies; the study is aimed at a non-conventional model to support dynamic changes.

The papers [8] (presented in Chapter 2.6) and [6] (Chapter 2.7) describe the use of a chemical metaphor to model service selection in compositions of services. The proposed model clearly distinguishes the selection of services from their execution, making it possible to specify user requirements for an SBA, and the QoS of the services offered by the providers in the same formalism through chemical molecules. Both user requirements and QoS refer to non functional characteristics of services/applications such as cost, time to deliver, and so on. Service offers are selected by chemical rules according to their QoS in order to obtain a concrete workflow that meets user requirements. In this respect, chemical rules implement strategies/policies that can be specified at the SCC level. The evolutionary nature of the proposed chemical mechanism allows to account for changes in strategies and policies through the insertion/activation of new chemical rules during the selection process. Furthermore, it allows also to change both user requirements and service offers with the insertion/deletion of chemical molecules as perturbations of the chemical system, so providing different concrete workflows when different conditions occur.

The work presented in [6] extends the mechanism presented in [8] by supporting the late binding of services at execution time. In fact, the chemical-based selection mechanism may produce service mappings with unresolved branches that can be instantiated at run-time if the execution requires it. With this approach the service selection process may be interleaved with the concrete workflow execution. So, changes in strategies/policies, and/or in the set of available service offers can be taken into account when the unresolved branches have to be executed.

Table 2.1: Contributed papers

Ref.	Paper	Partners	Research Challenges	IRF Elements
Sec. 2.1, App. A	A. Kertész, G. Kecskeméti, I. Brandic, "Autonomic SLA-aware Service Virtualization for Distributed Systems", In proceedings of the 19th Euromicro International Conference on Parallel, Distributed and Network-Based Computing, Ayia Napa, Cyprus, February, 2011. [15]	SZTAKI, TUW	Multi-level and self adaptation (2.3), Deployment and Execution Management (2.3), Proactive SLA negotiation and agreement (1.3), Comprehensive and integrated adaptation and monitoring principles, techniques, and methodologies (1.2)	*Conceptual Research Framework*: SI *Reference Life-Cycle*: Deployment & Provisioning, Operation & Management *Logical Run-Time Architecture*: Adaptation Engine
Sec. 2.2, App. B	J. Ejarque, A. Micsik, R. Sirvent, P. Pallinger, L. Kovács, R. M. Badia: Job scheduling with license reservation: a semantic approach. Proceedings of the 19th Euromicro International Conference on Parallel, Distributed and Network-Based Computing, PDP 2011 [11]	SZTAKI, UPC (assoc. partner)	Deployment and execution management, Multi-level and self-adaptation	*Logical Run-Time Architecture*: Resource Broker, Run-time QA Engine, Service Container *Conceptual Research Framework*: QA Capabilities, Service Infrastructure *Reference life-cycle*: Operation & Management, Deployment and Provisioning
Sec. 2.4, App. C	A. Lage Freitas, N. Parlavantzas, and J.-L. Pazat. 2010. A QoS assurance framework for distributed infrastructures. In Proceedings of the 3rd International Workshop on Monitoring, Adaptation and Beyond (MONA '10). ACM, New York, NY, USA, 1-8. [14]	INRIA	Multi-level and self-adaptation, Deployment and execution management	*Logical run-time architecture*: Adaptation engine *Conceptual Research Framework*: Service Infrastructure, Adaptation and monitoring, *Reference life-cycle*: Identify adaptation need, Identify adaptation strategy, Enact adaptation, Operation & Management, Deployment and Provisioning
Sec. 2.3, App. D	C. Di Napoli, M. Giordano: Experimenting Scheduling Policies with Continuation-based Services. In Proceedings of Int. Symposium on Advanced Topics on Information Technologies and Applications ITA 2010 (in conjunction with CIT2010), IEEE Computer Society Press, pp. 1498-1503, [9]	CNR	Deployment and Execution Management (2.3) Multi-level and self-adaptation (2.3)	*Conceptual Research Framework*: SI *Reference Life-Cycle*: Deployment & Provisioning, Operation & Management *Logical Run-Time Architecture*: Service Container

Table 2.1: Contributed papers

Ref.	Paper	Partners	Research Challenges	IRF Elements
Sec. 2.5, App. E	M. Caeiro, Zs. Németh, T. Priol: A chemical model for dynamic workflow coordination. Proceedings of the 19th Euromicro International Conference on Parallel, Distributed and Network-Based Computing, Ayia Napa, Cyprus, 2011, IEEE Computer Society	SZTAKI, INRIA, external	Multi-level and self-adaptation (2.3) Formal Models for QoS-Aware Service Compositions (2.2)	*Logical Run-Time Architecture*: Adaptation Engine *Reference life-cycle*: Enact Adaptation *Conceptual Research Framework*: SI, SCC *Logical Desing Environment*: Modeling Techniques
Sec. 2.6, App. F	C. Di Napoli, M. Giordano, Zs. Németh, N. Tonellotto: Using Chemical Reactions to Model Service Composition. In Proceedings of Second Int. Workshop on Self-Organizing Architectures (SOAR 2010) (in conjunction with ICAC 2010), pp. 43-50, ACM Press [8]	CNR, SZTAKI	Multi-level and self-adaptation (2.3) QoS-aware adaptation of Service Composition (2.2)	*Conceptual Research Framework*: SI, SCC, SAM *Reference Life-Cycle*: Enact Adaptation *Logical Run-Time Architecture*: Adaptation Engine
Sec. 2.7, App. G	C. Di Napoli, M. Giordano, Zs. Németh, N. Tonellotto: Adaptive instantiation of service workflows using a chemical approach. In Proceedings of CoreGRID/ERCIM Workshop on Grids, Clouds and P2P Computing (in conjunction with EuroPAR 2010), LNCS series, Springer, to appear [6]	CNR, SZTAKI	Multi-level and self-adaptation (2.3) QoS-aware adaptation of Service Composition (2.2)	*Conceptual Research Framework*: SI, SCC, SAM *Reference Life-Cycle*: Enact Adaptation *Logical Run-Time Architecture*: Adaptation Engine

Table 2.2: Contributed papers by Research Challenge/IRF Elements

Research Challenges	View:IRF Elements	Paper Acronyms
Multi-level and self-adaptation (2.3)	CRF:SI	[8], [6], [9], [7], [11], [15] , [14]
	CRF:SCC	[8], [6], [7]
	CRF:SAM	[8], [6]
	CRF:QAC	[11]
	LRTA:AE	[8], [6], [7], [15], [14]
	LRTA:RB	[11]
	LRTA:RQAE	[11]
	LRTA:SC	[9], [11]
	LDE:MT	[7]
	RLC:EA	[8], [6], [7], [14]
	RLC:OM	[9], [9], [11], [15], [14]
	RLC:DP	[9], [11], [15]
	RLC:IAN	[14]
	RLC:IAS	[14]
Deployment & Execution Management (2.3)	CRF:SI	[9], [?], [11], [15], [14]
	CRF:QAC	[11]
	LRTA:AE	[15], [14]
	LRTA:SC	[9]
	LRTA:RB	[11]
	LRTA:RQAE	[11]
	LRTA:SC	[11]
	RLC:OM	[9], [?], [11], [15], [14]
	RLC:DP	[9], [11], [15]
	RLC:IAN	[14]
	RLC:IAS	[14]
QoS-aware Adaptation of Service Composition (2.2)	CRF:SI	[6], [8]
	CRF:SCC	[6], [8]
	CRF:SAM	[6], [8]
	LRTA:AE	[6], [8]
Formal Models for QoS-Aware Service Compositions (2.2)	CRF:SI	[7]
	CRF:SCC	[7]
	LDE:MT	[7]
	LRTA:AE	[7]

Table 2.2: Contributed papers by Research Challenge/IRF Elements

Research Challenges	View:IRF Elements	Paper Acronyms
	RLC:EA	[7]
Proactive SLA negotiation and agreement (1.3)	CRF:SI	[15]
	LRTA:AE	[15]
	RLC:OM	[15]
	RLC:DP	[15]
Comprehensive and integrated adaptation and monitoring principles, techniques, and methodologies (1.2)	CRF:SI	[15]
	LRTA:AE	[15]
	RLC:OM	[15]
	RLC:DP	[15]

Abbreviations	Research Challenges and IRF Elements
CRF:SI	*Conceptual Research Framework*: Service Infrastructure
CRF:SCC	*Conceptual Research Framework*: Service Composition and Coordination
CRF:SAM	*Conceptual Research Framework*: Service Adaptation and Monitoring
CRF:QAC	*Conceptual Research Framework*: Quality Assurance Capabilities
LRTA:AE	*Logical Run-Time Architecture*: Adaptation Engine
LRTA:RB	*Logical Run-Time Architecture*: Resource Broker
LRTA:RQAE	*Logical Run-Time Architecture*: Run-time Quality Assurance Engine
LRTA:SC	*Logical Run-Time Architecture*: Service Container
LDE:MT	*Logical Desing Environment*: Modeling Techniques
RLC:EA	*Reference life-cycle*: Enact Adaptation
RLC:OM	*Reference life-cycle*: Operation & Management
RLC:DP	*Reference life-cycle*: Deployment & Provisioning
RLC:IAN	*Reference life-cycle*: Identify adaptation need
RLC:IAS	*Reference life-cycle*: Identify adaptation strategy

Table 2.3: Abbreviations of Research Challenge/IRF Elements used in Table 2.2

2.1 Autonomic SLA-aware Service Virtualization for Distributed Systems (SZTAKI+TUW)

In the S-Cube deliverable CD-JRA-2.3.2 [1] we introduced a conceptual architecture incorporating three closely related areas: agreement negotiation, service brokering and service deployment. This collaboration fits into the third research thread called Runtime Environment of the WP-JRA-2.3 Research Architecture, and to the Service Infrastructure element of the S-Cube Integrated Research Framework. We examined this architecture and revealed the basic requirements for a future self-* realization of these core components. These basic requirements implied 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. In deliverable CD-JRA-2.3.4 [2] we refined this vision, and presented a unified service virtualization environment representing the first attempt to combine SLA-based resource negotiations with virtualized resources in terms of on-demand service provision resulting in a holistic virtualization approach.

In this current contribution we examine a refined architecture considering resource provision using a virtualization approach and combining it with the business-oriented utilization used for the SLA agreement handling. Thus, we provide an SLA-coupled infrastructure for on-demand service provision based on SLAs (called SLA-based Service Virtualization – SSV). We also exemplify how autonomic behaviour appears in the architecture in order to cope with changing user requirements and on demand failure handling. Autonomic Computing is one of the candidate technologies for the implementation of SLA attainment strategies. Autonomic systems require high-level guidance from humans and decide, which steps need to be done to keep the system stable [4]. Such systems constantly adapt themselves to changing environmental conditions. Similar to biological systems (e.g. human body) autonomic systems maintain their state and adjust operations considering changing components, workload, external conditions, hardware and software failures. An important characteristic of an autonomic system is an intelligent closed loop of control.

The Autonomic manager in the SSV architecture is an abstract component that specifies how self-management is carried out. The manager can be implemented in the different layers of the architecture in order to handle the different unexpected situations. For the identification of failures we use case-based reasoning with a knowledge database, since the identification of failure sources (done by sensors) and mapping failures to appropriate reactions (done by actuators).

However the discussion of the knowledge database and layered notifications is out of scope of this paper, therefore here we mention three typical reactive actions we support in our SSV: namely negotiation bootstrapping, broker breakdown and self-initiated deployment. Negotiation bootstrapping occurs when the architecture needs to select a new negotiation strategy. Broker breakdowns are handled by the Meta-Broker component by initiating renegotiation on an already executed service call. Finally a service instance can guarantee the negotiated SLAs by deploying another service instance and redirecting the calls causing the unexpected service behaviour.

In order to evaluate our proposed SSV solution, we use a typical biochemical application as a case study called TINKER Conformer Generator application [5] using molecular modeling for drug development, which was gridified and tested on production Grids. The application generates conformers by unconstrained molecular dynamics at high temperature to overcome conformational bias then finishes each conformer by simulated annealing and energy minimization to obtain reliable structures. Its aim is to obtain conformation ensembles to be evaluated by multivariate statistical modeling. This use case can be regarded as a special, corresponding case of the S-Cube EHEALTH-BG-03 scenario called "Easier Planning of Examinations and Treatments". For the evaluation, we have created a general simulation environment, in which all stages of service execution in the SSV architecture can be simulated and coordinated in both Cloud and Grid environments. From the achieved results we ca n clearly see that the simulated SSV architecture using different distributed environments overperforms the former solution using only Grid resources. Comparing the different deployment strategies we can see that on demand deployment introduces some overhead, but service duplication can enhance the performance and help to avoid SLA violations with additional virtual machine deployment costs.

This work is closely related to the research topics on SLA-negotiation in WP-JRA-1.3 and on monitoring and adaptation in WP-JRA-1.2. More adaptation-related capabilities of the architecture (including cross-layer adaptation and SLA usage policies) will be reported in the upcoming deliverable of WP-JRA-1.2 (CD-JRA-1.2.5). This contribution addresses the following research challenges of the Integrated Research Framework: "Proactive SLA negotiation and agreement", "Comprehensive and integrated adaptation and monitoring principles, techniques, and methodologies" and "Deployment and execution management".

2.2 Job Scheduling with License Reservation: A Semantic Approach (SZTAKI+UPC)

Services often use licensed software internally during the execution of service requests. Similarly, Cloud and Grid environments also need to handle the issue of software licensing. At the end, this issue appears as a set of deployment and scheduling problems in SBAs; where can they run the licensed software, how many instances can be executed simultaneously, how to distribute licenses on various deployment points, etc. In this work we propose a semantic approach for scheduling and managing jobs by service and infrastructure providers taking into account the different software license issues. A model is given in the paper to describe software licenses similarly to other hardware and software resources. In this way, licenses can be adaptively allocated to the different requested jobs as another kind of resource according to their properties (license terms). Then, various scheduling approaches can be applied on the semantic description of requirements for service/job execution.

Software licenses are key components of distributed processing, especially the planning of the execution of long-running, computation-intensive workflows often used in Grid and Cloud environments. The lack of enough licenses can block the execution of a workflow similarly to the lack of appropriate computing resources. However, this topic is still rather undeveloped today. One cause for this may be the huge variety of license conditions, which makes the complete conceptualization of software licenses almost impossible.

In this paper a practical approach is taken; firstly, we only work with a subset of license content minimally required for job scheduling, secondly, we provide a pragmatic but extensible core for the semantic description of software licenses, and thirdly we limit the semantic descriptions to the viewpoint of resource management.

Licenses can be seen as part of the quality attributes of service-based applications. License requirements are similar to security requirements; according to the S-Cube quality reference model, they can be categorized as unmeasurable but controllable attributes on the infrastructure level within the service provider domain.

The paper belongs to work package JRA-2.3 as we cover a deployment and provisioning problem on the infrastructure layer, however the methodology and the semantic approach addresses work package JRA-1.3 as well.

Regarding WP-JRA-2.3 research challenges the paper is related to "Supporting adaptation of service-based applications" and "Runtime SLA Violation Prevention". The presented solution is about a semantic approach for the management of licenses bound to service executions. This approach can help to avoid violation of license agreements and also makes scheduling and adaptation more efficient regarding the use of licensed software.

2.3 A QoS assurance framework for distributed infrastructures (INRIA)

This paper proposes a framework to assist service providers in enforcing SLAs for services deployed on large-scale distributed infrastructures, such as clusters, grids, and clouds. Enforcing SLAs for such services is complex because of fluctuating service workloads and unpredictable resource availability. Accommodating this dynamism requires continuous monitoring of the operating conditions and performing corrective actions, such as re-allocating resources, migrating computational elements or re-negotiating agreements. The goal is to avoid violations of SLA terms while satisfying service provider-specific constraints, such as maximizing resource utilization.

The proposed framework, called QU4DS, simplifies implementing QoS assurance through a rich set of reusable QoS management mechanisms, including negotiation, translation and resource provisioning. Importantly, the framework supports dynamic adaptation; that is, support for monitoring and automatically modifying service behavior and resource usage. Service implementations and underlying resources are managed through a consistent, uniform infrastructure API, which minimizes the frameworks dependence on specific platforms and increases its applicability. A prototype of the framework has been developed building on the XtreemOS grid platform.

The paper includes a case study based on a service encoding audio files; the service implementation relies on the Master/Worker pattern. Initial experimental results demonstrate that QU4DS effectively reduces SLA violations in this case study. Specifically, the experiments analyze system behavior with QU4DS applying the "Single Replacement for Late Jobs" adaptation strategy in order to deal with faults. The experiments show that QU4DS can reduce by half SLA violations in comparison to applying no adaptation.

The work presented in the paper addresses the Multi-level Adaptation research challenge of WP2.3, since it proposes a framework for building self-adaptable services. Moreover, it addresses the "Deployment and execution management" challenge since the framework manages the deployment of multiple service instances on a shared infrastructure. The work answers the research question "Self-optimization and self-healing of a single service" as the framework concentrates on QoS assurance for individual services, enabling them to continue to conform to their SLA while making efficient use of underlying resources. Finally, the work is related to the question ”Runtime SLA Violation Prevention” as the included case study demonstrates that the framework can be used to prevent SLA violations.

2.4 Experimenting Scheduling Policies with Continuation-based Services (CNR)

The paper proposes a scheduler module as a component of a service provider infrastructure able to control at application level the execution of services and the corresponding resource allocation (in terms of CPU time). The control of service execution/resource allocation is realized through the use of *continuations* [16] to implement service execution suspension and resuming. The paper describes the implementation of two cost-driven resource sharing scheduling policies using the proposed continuation-based control mechanism. The policies take into account user requirements expressed in terms of service cost, by deploying and executing services with priorities based on the current cost and the provider workload. Both policies take into account that services submitted with a required QoS, i.e. with an associated cost, have to be executed with a priority depending on its value, but they differ in the degree at which the high priority of some services affects the execution of services with lower priority.

Simulations are reported in the paper to test the two different scheduling policies under different conditions that concern the provider's workload, the distribution of incoming requests, the arrival of requests changing the cost associated to the service. The experimental results show that the priority assigned by the scheduler to a service reduces the elapsed execution time according to the service cost and the workload, as expected. Furthermore, simulations show the degree at which the two policies react to service priority changes due to cost-update requests received by the provider during service execution.

The main result of this work is that the proposed continuation-based service scheduler is able to adapt the priority with which the service is executed to both the provider current workload and to changes in service costs that may occur during service executions. In order to support the possibility to change service costs during its execution, the service provider infrastructure is equipped with interfaces that can be invoked by service consumers to update their requirements. The use of continuations to control service execution provides a flexible mechanism to specify different scheduling policies at application level.

The work addresses some of the problems presented in the Deployment and Execution Management research challenge whose focus is the on-demand and dynamic service provisioning. In fact, the proposed application-level continuation-based scheduling mechanism allows to change at run-time parameters affecting service provision either driven by consumer or system requirements.

In the context of the S-Cube infrastructure, the continuation-based mechanism is a viable approach to take into account changes in the service requirements coming from both the BPM and SCC layers, and to adapt service provisioning on demand. In fact, the interfaces defined to control service requirements could be invoked by the user consumer (as shown in the paper), but also by other software components of the S-Cube infrastructure. With this respect, the work addresses also the research question “Self-optimization and self-healing of a single service” of WP2.3 by allowing service execution suspension and resuming on demand to re-configure the resources assigned to the service execution.

2.5 A chemical model for dynamic workflow coordination (SZTAKI+INRIA)

The work presented in [7] is complementary to other "chemical" papers, [6] and [8]. The chemical model for dynamic workflow coordination [7] is a different, yet related aspect to chemical based service composition. In fact, workflow is just a proper example to study the potential of a non-conventional, chemical modeling approach where enactment of scientific workflows in an ever-changing (with respect to performance and availability) and fault prone distributed heterogeneous computing environment requires a complex coordination. Available information may be partial or not up to date and the presence of advanced workflow patterns adds further uncertainty. We assume that such coordination requires frequent and abrupt changes where no predefined or static approaches are feasible. The same problems are relevant in other settings, e.g. service composition.

Workflows may change during execution due to any condition unforeseen at planning, user intervention, changes in the environment or for adaptation reasons. Changes may involve structural, functional, parametric changes; there may appear several versions and instances of the same workflow definition. The aim is to create a model that is flexible and allow establishing a fully dynamic enactment of workflow that can lead later to various adaptation strategies. We imagine a system that is able to react to any changes in its own behavior or the embedding conditions. A metaphor we took is chemistry: just like chemical reactions are self-governing, self-balancing, self-evolving processes, an "idealistic" distributed workflow enactment should possess many of these features. The contribution of [7] is a methodology by developing an abstract model for an on-the-fly dynamic assignment of activities and resources (service requests and service offers). The main considerations are: (i) support for a generalized notion of workflow graphs without restrictions; (ii) a clear separation and detailed model of data and control flow; (iii) modeling events and connectors to generalize control in the chemical model; (iv) support for dynamic structural and semantic changes by multiple workflows / multiple versions / multiple instances; (iv) detailed analysis of some fundamental constructs modeled as chemical entities and reactions.

The approach is orthogonal to that of presented in [6] and [8]. While the latter papers prepare a concrete workflow (service mapping) in advance, that (i) can be altered at runtime and (ii) contain unresolved branches that are instantiated lazily; [7] supports dynamicity by postponing decisions to the last moment and create the concrete workflow just-in-time. The difference stems from the assumed enactment model. This work is an additional study to [6] and [8] and hence, related to the Multi-level adaptation research challenge of WP2.3 yet, in a more restricted way. While [6] and [8] focuses on improving compositions, [7] is related to dynamic changes in the workflow structure.

2.6 Using Chemical Reactions to Model Service Composition (CNR+SZTAKI)

In the present contribution the process of selecting actual implementations of services composing an application required by an end-user is modelled as a chemical process. A user request is expressed as a set of molecules representing an Abstract Workflow that specifies the functionalities required for each component and their dependence constraints, together with some preferences expressed by the user on non-functional characteristics of the workflow (such as cost, time to deliver, and so on). For the time being, the proposed approach does not focus on specific non-functional characteristics of services since their specification depends on the application scenarios taken into consideration. The service implementations for each functionality are made available as *service offers* that specify both the service implementation end-point, and a value corresponding to a specific non-functional service characteristics (QoS) that drive the selection of a specific offer. Also service offers are represented as molecules that react according to the dependencies specified by the Abstract Workflow and some constraints on the QoS of services, embedded in chemical rules. These rules accomplish the selection of the component services and they can be enabled/disabled/changed on demand. As such, the proposed mechanism allows to specify and to change strategies/policies embedded in the chemical rules that perform the selection of the component services.

The main result of this work is to model the process of instantiating the required functionalities with actual service implementations as an evolving and always running chemical process that can take into account the current state of the context when the composition is required. The evolutionary nature of the chemical system provides a form of adaptation since once compositions of services are computed with the available services, new compositions can be computed as soon as new services become available or the conditions of existing ones change. So the proposed mechanism is able to self-adapt to changes depending both on the specific capabilities of component services available at the time when the request is issued, and the environmental conditions in which the system is being executed.

The work presented in the paper addresses the “Multi-level Adaptation” research challenge of WP2.3, since the proposed mechanism takes into account changes in the set of services available to compose the SBA by modelling the continuous insertion/deletion of new molecules, representing service offers, during the selection process as “perturbations” of the chemical system. These perturbations are processed by the chemical rules and they may produce alternative compositions of services that were not available before, or that can better meet the user preferences.

Furthermore, the proposed chemical approach allows to account for different adaptation types by changing the chemical rule triggers dynamically. As such, the approach addresses the research question “Supporting adaptation of service-based applications” since it provides a mechanism able to react to changes in the QoS requirements coming from the other levels of the S-Cube infrastructure. For this reason the work is closely related to the research challenge “QoS-aware Adaptation of Service Composition” of WP2.2.

2.7 Adaptive instantiation of service workflows using a chemical approach (CNR+SZTAKI)

In this paper the chemical-based mechanism to select service implementations (presented in section 2.6) is extended in order to support the dynamic binding of services at run-time when the required functionalities are not yet available in the system. The extension consists in introducing molecules that act as *placeholders* for missing service implementations, and new chemical rules to aggregate workflow fragments containing service placeholders. The proposed extension allows to produce concrete workflows ready to be executed even though some services are not yet bound to actual implementations. The placeholders are a means to realize the late-binding of services when (and if) their actual implementations become available later on.

The main result of this work is the possibility to interleave the selection of service implementations with their execution, so providing the possibility to execute a workflow also when some of its functionalities are missing and they are not likely to be invoked. For example, when “if” statements are present in an Abstract Workflow, it is known that only one path will be actually executed, so it is possible to look for service implementations only for one path and to start the execution of the workflow also without instantiating the other path. If during the execution the instantiated path is taken, then the execution can continue successfully. Otherwise the execution can stop if the corresponding service implementation is still missing, or it can continue if, in the meantime, the placeholder has been bound by the chemical system that concurrently processes the new coming offers. The same mechanism could be used also in the case a failure of an already selected implementation occurs.

This paper is an extension of the paper reported in section 2.6, so it addresses aspects of both the "Multi-level Adaptation" research challenge of WP2.3 and the QoS Aware Adaptation of Service Compositions research challenge of WP2.2. The present contribution goes also in the direction of supporting adaptation at run-time, so addressing the research question "Supporting adaptation of service-based applications" of WP2.3. In fact, the late-binding of service implementations copes with another form of adaptation to changes in the service provisioning at the composition layer. The proposed extension allows for an eager execution of a partially instantiated workflows in order to take advantage of the possibility that service implementations missing at the selection phase become available later on during the execution. Even though not directly addressed in the paper, the late binding of services at run-time based on chemical rules opens the possibility to implement different strategies/policies, specified at the higher layers of the S-Cube infrastructure, that cope with the selection of service offers that become available only when the corresponding services have to be executed.

Chapter 3

Conclusions

This deliverable presented our results related to elements and aspects of infrastructure level mechanisms for specifying adaptation policies and strategies. These mechanisms are targeting the objectives of the “Infrastructure Mechanisms for the Run-Time Adaptation of Services” research thread of the work package. The proposed mechanisms intend to support and prepare the last deliverable of this research thread called CD-JRA-2.3.8“Specifications of policies and strategies for distributed and multi-level adaptation”.

The deliverable is a collection of scientific papers published in conference proceedings and organized along the research directions of WP-JRA-2.3. The papers all have been peer reviewed which ensures that the papers represent significant contributions to service-based system research and they demonstrate progress in the WP. 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 2. These research directions are focused around three main topics. First, we have applied the chemical metaphor in [6, 8, 7] to reveal the possibilities on combining the infrastructure and composition level adaptation strategies. Next, we have demonstrated that quality aware infrastructure [11, 15, 14] can improve schedulers, predict future SLA violations and increase the autonomous behavior in the various infrastructure level components. Finally, the deliverable also focused on new approaches on the adaptation of infrastructure level scheduling to allow the system (*i*) to handle software licenses as quality requirements [11] and (*ii*) to allow intermittent execution of long running service executions [9].

The papers presented in this deliverable proposed mechanisms that focused on localized components of service based infrastructures. Our future work towards the deliverable CD-JRA-2.3.8 will demonstrate that these mechanisms could efficiently participate and support future multi-level adaptation scenarios.

Bibliography

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

[2] Decision support for local adaptation. S-Cube deliverable CD-JRA-2.3.4, December 2009

[3] First Version of Integration Framework. S-Cube deliverable CD-IA-3.1.3

[4] J.O. Kephart, D.M. Chess. The vision of autonomic computing. *Computer* . 36:(1) pp. 41-50, Jan 2003.

[5] TINKER Conformer Generator workflow:
http://www.lpds.sztaki.hu/gasuc/index.php?m=6&r=12

[6] C. Di Napoli, M. Giordano, Zs. Németh, N. Tonellotto: Adaptive instantiation of service workflows using a chemical approach. In Proceedings of the EuroPar Workshops, Springer, to appear

[7] M. Caeiro, Zs. Németh, T. Priol: A chemical model for dynamic workflow coordination. Proceedings of the 19th Euromicro International Conference on Parallel, Distributed and Network-Based Computing, Ayia Napa, Cyprus, 2011, IEEE Computer Society, to appear

[8] C. Di Napoli, M. Giordano, Zs. Németh, N. Tonellotto: Using Chemical Reactions to Model Service Composition. In Proceedings of Second Int. Workshop on Self-Organizing Architectures (SOAR 2010) (in conjunction with ICAC 2010), pp. 43-50, ACM Press

[9] C. Di Napoli, M. Giordano: Experimenting Scheduling Policies with Continuation-based Services. In Proceedings of Int. Symposium on Advanced Topics on Information Technologies and Applications ITA 2010 (in conjunction with CIT2010), IEEE Computer Society Press, pp. 1498-1503

[10] J. Ejarque, A. Micsik, R. Sirvent, P. Pallinger, L. Kovacs, R. M. Badia: Semantic Resource Allocation with Historical Data Based Predictions. The First International Conference on Cloud Computing, GRIDs, and Virtualization, CLOUD COMPUTING 2010, November 21-26, 2010 - Lisbon, Portugal

[11] J. Ejarque, A. Micsik, R. Sirvent, P. Pallinger, L. Kovacs, R. M. Badia: Job scheduling with license reservation: a semantic approach. Proceedings of the 19th Euromicro International Conference on Parallel, Distributed and Network-Based Computing, Ayia Napa, Cyprus, 2011, IEEE Computer Society, to appear

[12] P. Leitner, A. Michlmayr, F. Rosenberg, S. Dustdar: Monitoring, Prediction and Prevention of SLA Violations in Composite Services. In Proccedings of the IEEE International Conference on Web Services (ICWS'10), 2010

[13] I. Brandic, V. C. Emeakaroha, M. Maurer, S. Ács, A. Kertész, G. Kecskeméti, S. Dustdar: LAYSI: A Layered Approach for SLA-Violation Propagation in Self-manageable Cloud Infrastructures. The First IEEE International Workshop on Emerging Applications for Cloud Computing (CloudApp 2010), 34th Annual IEEE International Computer Software and Applications Conference Seoul, Korea

[14] A. Lage Freitas, N. Parlavantzas, and J.-L. Pazat: A QoS assurance framework for distributed infrastructures. In Proceedings of the 3rd International Workshop on Monitoring, Adaptation and Beyond (MONA '10). ACM, New York, NY, USA, 1-8.

[15] A. Kertész, G. Kecskeméti, I. Brandic: Autonomic SLA-aware Service Virtualization for Distributed Systems, Proceedings of the 19th Euromicro International Conference on Parallel, Distributed and Network-Based Computing, Ayia Napa, Cyprus, 2011, IEEE Computer Society, to appear

[16] D. P. Friedman, C. T. Haynes, and E. E. Kohlbecker. Program Transformation and Programming Environments. ch. Programming with Continuations, pages 263–274. Springer, 1984.