



Interactive Real-time Multimedia Applications on Service Oriented Infrastructures

ICT FP7-214777

White Paper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia

IRMOS_ White_Paper_v1.0

Scheduled Delivery: 31.08.2009
Actual Delivery: 27.02.2010
Version 1.0

Project co-funded by the European Commission within the 7th Framework Programme		
Dissemination Level		
PU	Public	
PP	Restricted to other programme participants (including the Commission)	
RE	Restricted to a group specified by the consortium (including the Commission)	X
CO	Confidential, only for members of the consortium (including the Commission)	

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	



Responsible Partner: ICCS/NTUA

Revision history:

Date	Editor	Status	Version	Changes
20.07.2009	Dimosthenis Kyriazis	TOC	0.1	Table of Contents & Document Formatting
10.08.2009	Dimosthenis Kyriazis	Draft	0.2	Updated Table of Contents
29.09.2009	Dimosthenis Kyriazis	Draft	0.3	Content Outline in each section
23.11.2009	Dimosthenis Kyriazis	Draft	0.4	Updated outline based on WP Leaders comments
18.12.2009	Dimosthenis Kyriazis	Draft	0.5	Initial Content in all sections
05.02.2010	Dimosthenis Kyriazis	Draft	0.6	Version for internal QA
27.02.2010	Dimosthenis Kyriazis	Final	1.0	Final Version

Authors

Alessandro Mazzetti (GILABS), Soeren Berger (USTUTT), Michael Braitmaier (USTUTT), Michael Boniface (IT-Inn), Tommaso Cucinotta (SSSA), George Kousiouris (NTUA), Dimosthenis Kyriazis (NTUA), Sai Narasimhamuthy (XY), Andreas Menychtas (NTUA), Karsten Oberle (ALUD), Bassem Nasser (IT-Inn), Thomas Voith (ALUD), Eduardo Oliveros (TID)

Internal Reviewers

Georgina Gallizo (USTUTT), Malcolm Muggeridge (XY)

Copyright

This report is © by the IRMOS Consortium 2008 - 2010. Its duplication is allowed only in the integral form for anyone's personal use and for the purposes of research or education.

Acknowledgements

The research leading to these results has received funding from the EC Seventh Framework Programme FP7/2007-2011 under grant agreement n° 214777

More information

The most recent version of the public deliverables of IRMOS can be found at <http://www.irmosproject.eu>

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

Glossary of Acronyms

Acronym	Definition
AC	Application Component
ASC	Application Service Component
ASCD	Application Service Component Description
A-SLA	Application Service Level Agreement
ANN	Artificial Neural Network
CPU	Central Processing Unit
D	Deliverable
DB	Database
DES	Discrete Event Simulation
DoW	Description of Work
EC	European Commission
EE	Execution Environment
fps	Frames per second
FP7	Framework Programme 7
FSM	Finite State Machine
ISONI	Intelligent Service Oriented Network Infrastructure
IXB	ISONI eXchange Box
MDS	Monitoring and Discovery System
OS	Operating System
PH	Physical Host
PM	Project Month
PU	Public
QA	Quality Assurance
QoS	Quality of Service
PES	Performance Estimation Service
RE	Restricted
RT	Real-time
SC	Service Component
SLA	Service Level Agreement
SOA	Service Oriented Architecture
T-SLA	Technical Service Level Agreement
Technical WPs	WP4, WP5, WP6, WP7
UML	Unified Modeling Language
VMU	Virtual Machine Unit
VSN	Virtual Service Network
VSND	Virtual Service Network Description
WP	Work Package
WS	Web Service

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

Table of Contents

1. Executive Summary.....	5
2. Introduction	6
2.1. IRMOS & Cloud computing	6
2.2. Real-time in Clouds.....	7
3. The IRMOS Approach	8
3.1. Vision	8
3.2. Architecture	10
4. Achieving Real-time across the IRMOS Layers.....	11
4.1. Outline.....	11
4.2. Enabling Real-time.....	13
4.2.1. Service Engineering	13
4.2.2. Negotiation	15
4.2.3. Reservation.....	17
4.3. Real-time	18
4.3.1. Execution and Monitoring.....	19
4.3.2. Re-negotiation.....	21
5. Conclusions.....	24
6. References.....	25

List of Figures

Figure 1: Mapping IRMOS to the Cloud	6
Figure 2: IRMOS Platform Phases.....	9
Figure 3: IRMOS Simplified Architecture	10
Figure 4: IRMOS Control Loops	12

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

1. Executive Summary

The advancements in distributed computing have driven the emergence of service-based infrastructures that allow for on-demand provision of ICT assets. Taking into consideration the complexity of distributed environments, significant challenges exist in providing and managing the offered on-demand resources such as compute, storage and network with the required level of Quality of Service (QoS), especially for real-time interactive and streaming applications.

The primary objective of the EU IRMOS Project is to develop cloud solutions that support real-time QoS guarantees for interactive multimedia applications by providing coherent and consistent real-time attributes at various levels of the infrastructure (application, network, storage, processing). The architecture considers the full lifecycle of service-based systems deployed on cloud resources including service engineering, Service Level Agreement (SLA) negotiation and management, service provisioning and monitoring. QoS parameters at application, platform and infrastructure levels are given specific attention as the basis for provisioning policies in the context of temporal constraints. The goal of this whitepaper is to describe how real-time aspects are addressed in the architecture and thus providing the whole picture across all IRMOS layers.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

2. Introduction

2.1. IRMOS & Cloud computing

Today, many architectural paradigms from distributed computing such as service-oriented infrastructures, Grids and virtualisation are incorporated into “Clouds”. Research in different fields has driven the emergence of new classes of service-based systems called Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). IRMOS architecture concepts are directly aligned with the current cloud architecture, even though the IRMOS Project was initiated prior to the general acceptance of the cloud computing stack:

- *Infrastructure as a Service (IaaS)*, which refers to the provision of ‘raw’ machines (servers, storage, networking and other devices) on which the service consumers deploy their own software (usually as virtual machine images). In IRMOS, this functionality is provided by the Intelligent Service Oriented Network Infrastructure (ISONI) [4].
- *Platform as a Service (PaaS)*, which refers to the provision of a development platform and environment providing services and storage, hosted in the cloud. In IRMOS, this functionality is provided by Service Management and Service Engineering tools [5].
- *Software as a Service (SaaS)*, which refers to the provision of an application as a service over the Internet or distributed environment. In IRMOS, this functionality refers to the adaptation of the application to be able to run in this environment, for which a set of tools and a specific methodology have been developed [2].

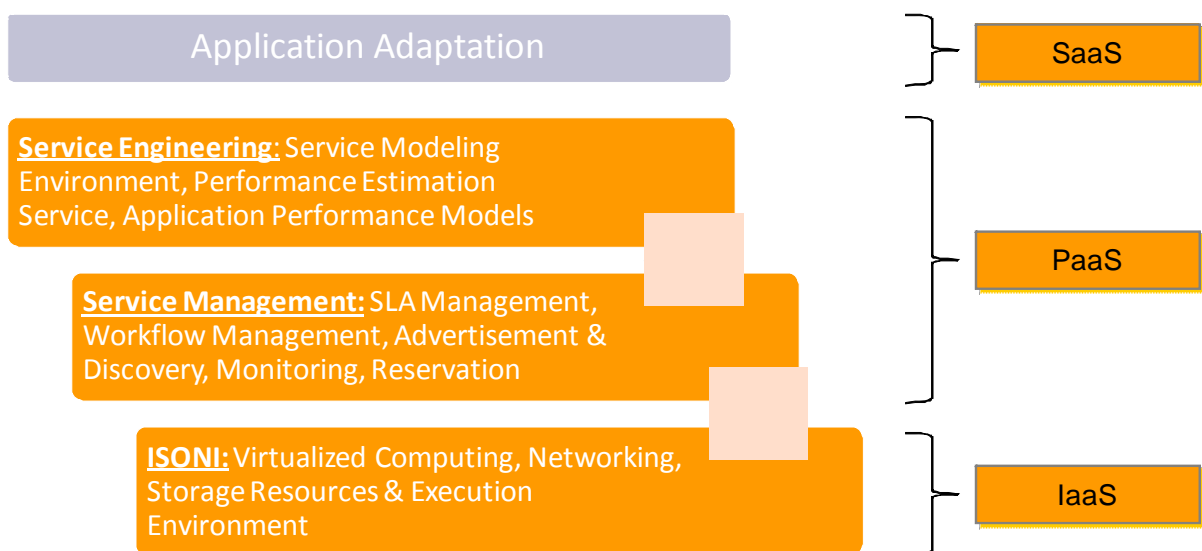


Figure 1: Mapping IRMOS to the Cloud

In the following chapters we describe how real-time aspects are addressed across the cloud by the services, tools and methodology developed by the IRMOS project.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

2.2. Real-time in Clouds

Before discussing the challenges for deploying and executing real-time interactive applications on the cloud, let us clarify the term “real-time”. Traditionally, “real-time” refers to hard real-time systems, where even a single violation of the desired timed behaviour is not acceptable, for example because it leads to total failure, possibly causing loss of human lives. However, there is also a wide range of applications that also have stringent timing and performance requirements, but for which some violations of the timing constraints are acceptable provided these are well understood and carefully managed, as they lead to degradation in the provided QoS level. These are “soft real-time” applications and include a broad range of interactive and collaborative tools and environments, including concurrent design and visualization in the engineering sector, media production in the creative industries, and multi-user virtual environments in education and gaming. In this paper, we focus on interactive soft real-time applications where one or more users interact with the application and with each other.

IRMOS has identified the main challenges for deploying and executing real-time interactive applications on the “Cloud” are summarized as follows:

- *Application Modelling*: techniques for modelling, predicting and estimating the application’s resource needs, which are expressed through models and artefacts capturing the application requirements.
- *Commitments and Obligations*: *business processes for establishing legal Service Level Agreement contracts* covering the relationships different domains in respect to PaaS, SaaS and IaaS providers. Given the dynamic nature of interactive real-time applications, automated and on-demand SLA negotiation is needed.
- *Resource management*: techniques to manage and assign precise portions of resources to real-time interactive applications. Usually, a soft real-time capable system guarantees a stable allocation of resources to individual applications at the cost of a slightly decreased overall system throughput/performance.
- *Performance monitoring and management*: mechanisms and processes for failure detection, monitoring and recovery in order to guarantee failsafe execution of the application on the required QoS level. This is also common for the non real-time application with the exception that in this case the time frame for the detection and reaction from abnormal situation is limited.
- *Converged Real-time Infrastructure*: the individual mechanisms that need to be combined for the provisioning of QoS levels for the various types of available resources: computing (performance guarantees for different parameters, hardware types and architectures), storage (guarantees for different parameters such as delay, latency, jitter, load balancing, etc) and networking (guarantees not only for ensured bandwidth but also for ensured delay and jitter for interactive media services like audio and video).

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

3. The IRMOS Approach

3.1. Vision

Service-oriented principles are an integral part of the IRMOS architecture. IRMOS adopts a service-oriented approach to allow services to interact dynamically and continuously, spanning between different domains, and ranging from the application level down to the level of network resources management and the execution environment.

The IRMOS project is developing the necessary infrastructure to allow multi-tenant hosted services and applications to run with predictable levels of QoS (e.g., in terms of throughput and latency) while allowing providers to share physical resources among the multitude of applications that will be instantiated at run-time. Current trends in resource management in distributed environments tend to focus on best-effort performance or service levels within the very limited scope of a single service or resource (e.g. compile farms or virtual machines), IRMOS advances the state-of-the-art in resource management by allowing users and providers to establish well-defined SLAs including performance terms which are guaranteed on an end-to-end basis. IRMOS combines computational nodes, network links and storage units, which are all capable of providing temporal guarantees to individual activities, while the underlying physical resources are shared across multiple applications and users.

The proposed architecture supporting real-time interaction between distributed set of people and resources requires the following key features:

- Real-Time QoS Specification: Language and associated toolkit for the specification of applications and application service components considering both structure and real-time QoS requirements.
- Performance Prediction: QoS oriented service engineering methodology and models for predicting performance metrics contingent to applications and resourcing events considering temporal profiles of application service components deployed on virtualised infrastructures.
- Dynamic SLA Negotiation and Management: SLA negotiation and management services supporting the dynamic negotiation (and re-negotiation) of Application-SLAs considering customer requirements and dynamic discovery of resources available, which fulfil these requirements (Technical-SLAs).
- On-Demand Resource Provisioning: Provisioning services for application service components on virtualised infrastructures through a combination of workflow and service-based management wrappers enhanced to support temporal profiles.
- Real-Time and QoS aware Scheduling: Capability of scheduling access to shared resources such as networks, storage and computing hardware in a way that limits temporal interferences among competing applications so as to preserve individual guarantees.
- QoS Monitoring: Services for measuring QoS at both application and infrastructure levels targeting trigger events for runtime adaptability of resource provisioning estimation and decision making.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

To achieve the real-time functionality and the required QoS level, the IRMOS platform operation is separated in two phases: the *offline*, where the application and Application Service Components (ASCs) are prepared (development, modelling, etc) and the *online*, where the resources are negotiated and reserved and the application is initialized and operate. Expanding on this in greater detail as depicted in the following figure (Figure 2):

- ***Offline Phase (design-time service engineering):*** This phase includes the processes for developing / adapting application components to the IRMOS environment and the creation of descriptors and documents for the application operation such as models, mapping rules, initialization scripts, SLA templates and workflow descriptions.
- ***Online Phase (negotiation, execution and monitoring):*** This phase begin with the SLA negotiation for both the Application and Technical SLA (as discussed in Section 4.2.2). As soon as the SLAs are agreed (signed), ISONI (as IaaS) reserves the resources (computational, storage and network) for use within the requested time interval. When the execution of the application starts, the Framework Services (as PaaS) are responsible for orchestrating and monitoring, until completion, the workflow execution. At any time during the execution, an exception and/or SLA violation occurs; mechanisms to adapt the resources (e.g. live migration) are put in place while re-negotiation of SLAs may be triggered in order to re-guarantee the QoS provision of the application and the application service components.

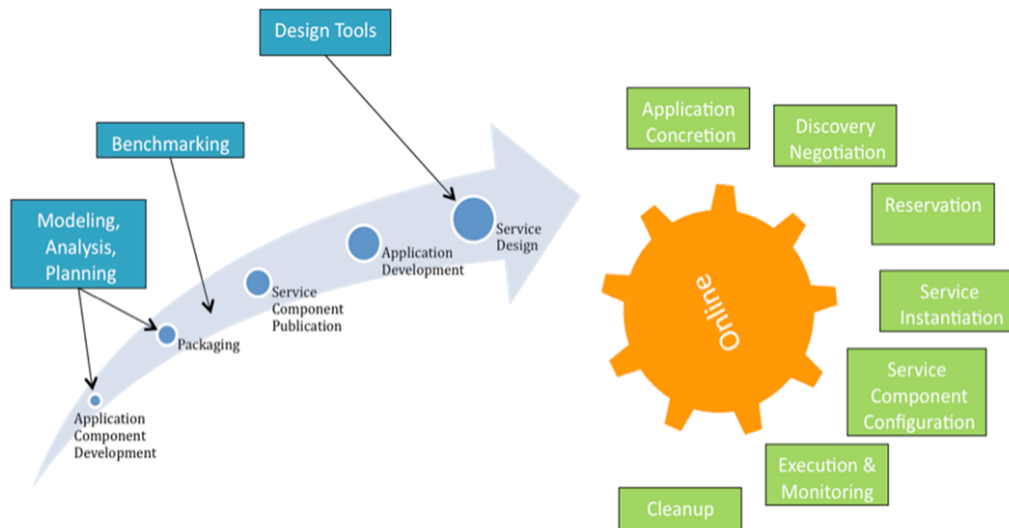


Figure 2: IRMOS Platform Phases

In general, real-time is required for the execution and monitoring processes intra and inter the Virtual Service Networks (VSNs) [4]. In that way, the real-time functionality spans from the platform modules that manage the resources (e.g. temporal isolation through scheduling capabilities) to the set of services (also known as framework services, e.g. workflow enactor) that are deployed with the application in the virtual environment.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

3.2. Architecture

Based on the cloud service models (as discussed in Section 2.1), in this section we briefly discuss the overall IRMOS Architecture (details can be found in [6]) in order to present the mapping of the IRMOS layers to the corresponding cloud service models we will then describe how real-time is achieved across these layers / cloud service models. In the **SaaS service model**, a specific methodology and tools have been developed, which allow application developers to engineer their application to deploy it within IRMOS. The **PaaS service model** refers to the IRMOS Framework Services (FS) layer; this operates between applications and virtualized resources. As shown in the following figure (Figure 3), the core elements are Service Engineering and Service Management, which are described in more detailed in the subsequent sections. The IRMOS FS layer aims to provide and manage the execution of real-time services inside the ISONI Environment (IaaS) on request of the Application Layer, while conforming to the real-time constraints as determined in the Application-SLA. Apart from managing applications execution, the framework supports service engineering, fully automated SLA negotiation and re-negotiation, mapping high level performance parameters to low level resource parameters, discovery and reservation of the ISONI resources needed for the execution. During the execution phase of the application, the FS monitor continuously and manage the application components and the resources either directly through the application wrappers or through the monitoring interface of the ISONI layer. The **IaaS service model** refers to ISONI, which is an infrastructure, consisting of a network of resources (computing, storage and networking). The infrastructure is managed and controlled by an ISONI middleware that allows resource sharing among multiple services (keynote of a cloud).

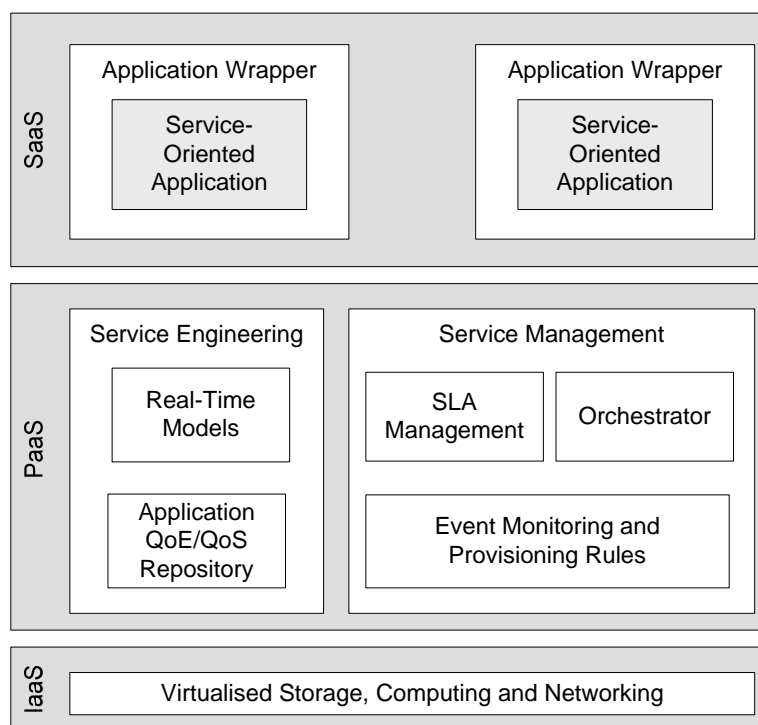


Figure 3: IRMOS Simplified Architecture

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

4. Achieving Real-time across the IRMOS Layers

4.1. Outline

In order to provide QoS guarantees for interactive real-time multimedia applications, using the IRMOS architectural approach we assume in this paper that the application developers have used the Service Engineering tools to engineer their applications for the IRMOS or any cloud platform (details of this process are provided in [2]). We now focus on the processes employed to support application provisioning and execution through the virtualised execution environment and networking infrastructure. To achieve this, the IRMOS platform does not merely provide a set of services but also cross layer workflows that consider the control channels and information exchanges which are required to support real-time management of interactive applications throughout the full lifecycle.

In order to minimize manual configuration and deliver on-demand QoS-aware services, all subsystems are self-managed and reconfigured in order to achieve management efficiencies, and to react on QoS failures (such as an SLA violation or network link failure) in a timely way. To achieve the latter, IRMOS introduces three control loops that are all at technical level and provide the necessary functionality in order to maintain QoS metrics across the architectural levels. The **IRMOS Control Loops** are the following and are depicted in Figure 4:

1. *Application Control*: It deals with the relationship between users and applications required to guarantee the application QoS. This control loop is managed by the application itself and the application developer in response to either user events or platform events. It is implemented with the use of models, workflows and tools that produce artifacts capturing the applications' behavior and estimating resource needs in advance of execution. During runtime it refers to application monitoring that may for example trigger events or require for changes in the provided resources.
2. *Environment Control*: It deals with the relationship between applications and virtual resources in order to guarantee the platform QoS, as agreed in the SLAs. This control loop is managed by the platform services in response to application and virtualisation events. It is implemented by the framework services (set of tools) that support and manage the applications at run-time (e.g. actions triggered if either the application or resources do not perform as expected or need to be adjusted).
3. *Virtualization Control*: It deals with the relationship between virtual and physical resources in order to guarantee the infrastructure QoS. This control loop is managed within ISONI in response to platform or physical events. It is implemented by intelligent networking services and tools as well as by the Execution Environment for computing and storage services.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

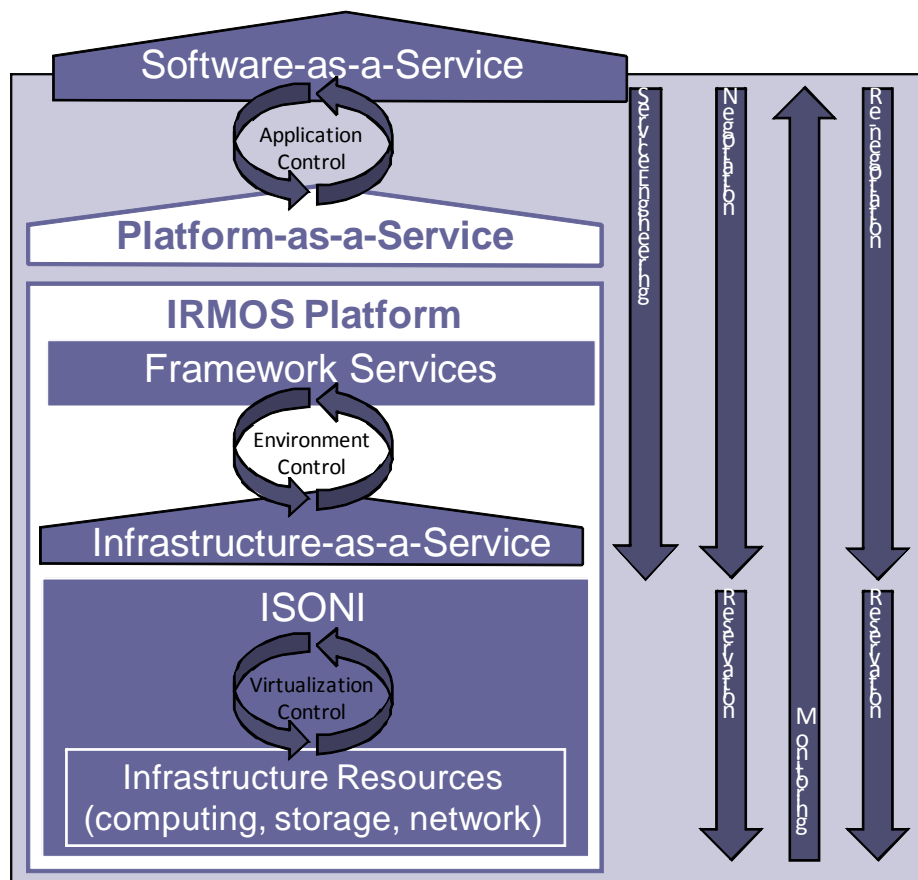


Figure 4: IRMOS Control Loops

The actual implementation of the control loops refers to tools and services used on different levels in order to monitor the applications' execution, communicate possible events and take corrective actions if needed. We focus on five (5) main processes / channels implementing the control loops, which we analyze in the following paragraphs of this whitepaper (while the first three are considered to be enablers for real-time, the latter two are "facilitators" of real-time and interactivity as explained later in this paper):

- Service Engineering
- Negotiation
- Reservation
- Monitoring
- Re-negotiation

It has to be noted that the Monitoring process is depicted as the only process / channel providing information and triggering actions originating from the infrastructure resources. However, the developed monitoring mechanism within IRMOS allows for monitoring of the applications as well. The latter is not captured in the above figure, since it shows how Control Loops can be applied in any cloud-based platform without the need of application monitoring on the Software-as-a-Service Layer.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

4.2. Enabling Real-time

The goal of the channels / processes described in this section is to provide the necessary information prior to execution in order to describe and model the applications, predict their behaviour, negotiate and reserve resources according to the aforementioned modelling and prediction in order to ensure that during execution the requested level of quality will be offered by the infrastructure.

4.2.1. Service Engineering

The **goal** of the Service Engineering process is to estimate the resources required for an application and identify the QoS parameters that have critical influence on the application's performance.

The **prerequisites** for the Service Engineering process are the Application Service Components (ASCs) Binaries, along with the Application's Configuration, Control & Monitoring Scripts. These are produced by the Application Developer.

The main **actor** in this process is the *Application Developer*, who uses a set of tools (namely Service Engineering Tools) to provide the necessary information to the platform services in order to estimate the required resources for the application execution. Furthermore, the *PaaS Provider* also participates during the Benchmarking process, which will be explained later.

The **tools / services** engaged in this process are the following:

1. *Service Modelling Environment*: A dedicated open source tool (Papyrus) that contains a profile for modelling ASCs using UML2 based on the Eclipse environment.
2. *Mapping Service*: A service providing an Artificial Neural Network-based rule/model, that depicts the relationships between the ASC characteristics/inputs, the different hardware configurations and the resulting QoS levels. It connects high level application workload features (such as number of users, resolution of processed images etc.) with application QoS requirements (like fps output, application response time etc) and low level resource parameters. Through these rules, the platform provider can observe the effect of selected resources on the QoS output for a given execution with specific workload parameters.
3. *Performance Estimation Service*: A service using the mapping rules in conjunction with modelling approaches such as Finite State Machines (FSMs) and Discrete Event Simulation (DES) in order to include workflow, events, interactivity, uncertainty and optimization in the ASC and application performance models, along with probabilistic guarantees.

The **flow of events** and the **outcomes** of this process are the following:

- The Application Developer uses the IRMOS Papyrus tool to model the application and specify a number of parameters that are necessary for the effective deployment of an ASC, and an application in general. These include workload parameters that affect the performance requirements of the ASC (e.g. number of

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

users connected to an eLearning application) and metrics that are used in order to quantify the level of QoS offered by the platform for this ASC (e.g. response time of the eLearning real-time server). To create a model of an application initially the interactions between several ASCs are examined following a standard UML technique. The whole process results in an application controller, its Finite State Machine (FSM) and the FSM for the entire application. IRMOS suggests an activity diagram of the application which shows how various ASCs interact. Further details on this process can be found in [5]. The outcomes of this process are the Application Service Component Description (ASCD) as well as the Application Description (connection of elementary ASCs in a workflow manner).

- The IRMOS Papyrus bundle automatically produces the Application-SLA Template by combining the individual ASCDs of the components. The Application-SLA is then published by the SaaS Provider to the SLA service of the PaaS Provider.
- The Application Developer benchmarks the performance of the application under different resource configurations. During this phase the developer provides a range of values for the input variables or characteristics used by the ASC. These values represent different use cases of the ASC and ideally the cases where the behaviour of the ASC alters. Each ASC is executed as a standalone component while in a normal execution it is part of a workflow, taking input for example from another ASC. In the benchmarking mode inputs are simulated so that the standalone execution is feasible. Resources are selected randomly and the effect of the selection is depicted in the QoS metric specified by each ASC. One very important characteristic that is also investigated is the hardware configuration with regard to real-time attributes, like the real-time scheduler used inside IRMOS. The ASC behaviour with regard to this kind of decision (such as granularity of CPU allocation) is investigated in order for this aspect to be included during the negotiation phase with the IaaS provider. The outputs used by this process are the QoS characteristics of the ASC (self-monitorable characteristics like response time in a request-response application, fps for the output video produced from a multimedia application etc).
- The Mapping Service receives a data set from the Monitoring database within the Framework Services (PaaS) detailing the behaviour of the ASC in relation to the input data parameters used during the execution of the specific ASC on a specific hardware platform (output of the benchmarking process). It then creates the ANN model of the ASC, connecting these high level input parameters and the hardware configuration used with the resulting QoS levels expressed in a specific ASC metric. The model is used for the run time estimation of the QoS parameters required. In this process, this service exploits the results of the benchmarking phase and produces the rules that will be accessible from the Performance Estimation Service for further processing.
- The Performance Estimation Service analyzes (PES) the rules created by the Mapping Service and uses further modelling techniques like FSM or DES in order to include interactivity, uncertainty and the dependencies between the different ASCs in a workflow. The outcome of this is to produce the necessary low level resource requirements (needed resources according to the estimation) for each ASC in the workflow.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

The functionality of the **IRMOS Control Loops** applied to the Service Engineering process refers to:

- *Application Control*: The Application Performance Models allow for modelling user demand in respect to application QoS. Optimization and updates of the models is feasible through a feedback loop that provides updated runtime information (received from the Monitoring service).
- *Environment Control*: The set of low level resource requirements that is being produced by the PES Service allows for modelling application QoS in respect to virtual resources. The updates that may occur on the application control loop can be “passed” to the environment control through the updated model or updated ASCD parameters that will be reflected in an updated set of the resource requirements.

4.2.2. Negotiation

The **goal** of the Negotiation process is to agree the Application and Technical SLAs between the customer and the corresponding providers in order to proceed with the reservation of the resources according to the agreements.

The **prerequisites** for the Negotiation process are the Application-SLA Template and the Rules (from the Mapping Service) as produced by the Service Engineering process.

There are many **actors** in this process, namely the *Customer*, the *SaaS Provider*, the *PaaS Provider* and the *IaaS Provider*, since there are two different negotiations involved: (i) Application-SLA negotiation between the *Customer* and the *SaaS Provider*, and (ii) Technical-SLA negotiation between *PaaS Provider* and *IaaS Provider*.

The **tools / services** engaged in this process are the following:

1. *IRMOS Portal*: A web interface being used by the Customer during the Application-SLA Negotiation Process.
2. *SLA Negotiator*: A service orchestrating the negotiation process and providing valid SLA offer to the Customer prior to the execution of the services. It represents the central component during the SLA negotiation process.
3. *A-SLA Manager*: A service responsible for the management of Application-SLAs, which includes query, publishing, creation and update SLA templates and mapping commitments to IaaS resources.
4. *Performance Estimation Service*: A service providing resource specification decisions, through the Virtual Service Network Description (VSND), which encompasses information related to the Virtual Machine Units and the network links interconnecting them. This information includes QoS annotations as requests towards the IaaS providers.
5. *Discovery Service*: A service responsible for registering available IaaS Providers that meet the QoS parameters defined in the Application-SLA. It is designed so as to store the pricelists of the IaaS providers and to retrieve them when contacted by the SLA Negotiator. The service also includes a function that is used by the IaaS providers to advertise their capabilities with QoS properties.
6. *T-SLA Manager*: A service responsible for the management of Technical-SLAs which specify resources procured with IaaS providers. Technical-SLAs are

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

offered by the IaaS providers as a response to requests from the PaaS providers. One of the key functionalities of the T-SLA Manager is reporting any SLA violations to the SaaS provider through a notification mechanism that can then be used to trigger events for mitigating management actions. When violation event occurs, the violation information will be sent out to the subscribed party.

7. *ISONI SLA Manager*: A service responsible for negotiating the Technical-SLA with PaaS Provider. If the Technical-SLA has been negotiated successfully, the ISONI SLA Manager launches the preparation and deployment by instantiating a VSN specific Deployment Manager instance inside ISONI (further details can be found in [7]).

The **flow of events** and the **outcomes** of this process are the following:

- The Customer initiates the process by using the IRMOS Portal to request Application-SLA templates (as produced by the Service Engineering Process) for a specific application. The IRMOS Portal requests from the SLA Negotiator a SLA template, which is edited by the Customer to define specific parameters that are essential for the application execution.
- The SLA Negotiator receives the price list of the resources from all IaaS Providers by inquiring the Discovery Service according to the terms included in the Application-SLA.
- The SLA Negotiator triggers the PES for each one of the available IaaS providers providing the Technical-SLA Template and the Application-SLA.
- The PES produces resource estimates (e.g. network bandwidth, CPU speed or storage latency) as well as the estimated price according to the IaaS provider price lists based on the requirements included in the Application-SLA as well as the rules produced by the Mapping Service during the Service Engineering Process. The translation of the high level requirements to the low level resource estimates results in a list of VSNDs sorted by cost.
- The SLA Negotiator sends the VSND to the T-SLA Manager who in turn sends an "Invitation To Treat" (Technical-SLA Template with desired contract information filled in, including the VSND) to the respective IaaS provider, more specifically to the ISONI SLA Manager, which checks the available resources and dependencies and makes a pre-reservation.
- The ISONI SLA Manager either formulates an Offer for the requested Virtual Service Network (VSN) or refuses to provide any Offer. The Offers returned by the IaaS Providers are already binding on said providers and, therefore, carry an expiration time so as to not make the IaaS Provider liable to guarantee the offered service indefinitely.
- The SLA Negotiator receives the Offers and refusals for the "Invitation To Treat", evaluates and prioritizes the received Offers according to the Customer's preferences.
- The SLA Negotiator creates the Technical-SLA and updates the Application-SLA with the reference to the specific Technical-SLA when an Offer is chosen.

The functionality of the **IRMOS Control Loops** applied to the Negotiation process refers to:

- *Application Control*: The requirements expressed in the Application-SLA allow for negotiation of SLAs and as a result reservation of resources according the

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

application's QoS requirements. Changes in the requirements are reflected in the Application-SLA and as a result in the selected resources.

- *Environment Control*: The Virtual Service Network Description (VSND) that is being produced by the PES Service actually turns the Application-SLA requirements into Technical-SLA requirements through the mapping of high-level terms to low-level resource estimates. Actual application and resource behaviour is reported by the PES (feedback loop), this is used to check whether the required QoS is actually supplied by the IaaS provider. This data is also used to validate the accuracy of models used by the engine by comparing the predictions with actual measurements.
- *Virtualization Control*: The Technical-SLA commitments allow for resource provisioning according to the application requirements expressed. Given that a pre-reservation takes place while creating the offers according to the VSND, the virtual resources provided meet the application's QoS. Changes in the infrastructure with regard to Technical-SLA commitments result in a domain wide resource availability check and a pre-reservation in the ISONI nodes for computing, storage and network resources.

4.2.3. Reservation

The **goal** of the Reservation process is to reserve the resources (virtual and physical) according to the agreed Technical-SLA between the PaaS and the IaaS provider. This process is part of the negotiation process described above. The main responsible **actor** for the ISONI (IaaS) internal reservation process is the ISONI Deployment Manager that is instantiated by the ISONI SLA Manager. The VSN specific and individual instance of the Deployment Manager has the responsibility to trigger and watch over the reservation process inside ISONI.

The **tools / services** engaged in this process are the following (some are not explained in detail since they are the same as in the Negotiation Process – Section 4.2.2):

1. *IRMOS Portal*
2. *SLA Negotiator*
3. *T-SLA Manager*
4. *ISONI SLA Manager*
5. *Deployment Manager*: A service responsible for the deployment of specific VSNs in the IaaS. It queries the domain level to checking for the resource availability. With it receives the node proposals it initiates the reservation process on the elected nodes.
6. *Resource Manager*: A service responsible for managing the execution resources (compute & storage) within the IaaS domain. The domain level is responsible for providing information about available computing resources. At the node level it reserves and allocates the computing resources.
7. *Storage Manager*: A service responsible for managing the reservation of storage resources within the IaaS domain.
8. *Path Manager*: A service responsible for managing the network resources within the IaaS domain. The domain level is responsible for providing information about available networking resources. The node level reserves and allocates the networking resources.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

The **flow of events** and the **outcomes** of this process are the following:

- Before a reservation can take place, the customer (through the IRMOS Portal) may need to upload the binaries (ASCs and corresponding scripts) to the PaaS Provider and submit specific configuration information through the Portal to the SLA Negotiator, which needs to be further uploaded to the ISONI (IaaS).
- The SLA Negotiator sends the reference to the Technical-SLA to the T-SLA Manager.
- The T-SLA Manager submits the binaries (link to the repository) to the ISONI SLA Manager along with the Technical-SLA.
- The ISONI SLA Manager commits the reservation of the resources (through the Resource, Path and Storage Managers [4]) and waits until the binaries are uploaded to the IaaS Provider. Afterwards an acknowledgement is sent to the T-SLA Manager. The outcomes of this process are the reserved resources.
- The Deployment manager deploys the VSN in the IaaS Provider domain (ISONI) including the Workflow Enactor and Monitoring instances that are part of the VSN.
- The T-SLA Manager requests for configuration information from the SLA Negotiator.
- The SLA Negotiator submits the configuration information to the Workflow Enactor instance that has been deployed in the IaaS.

The functionality of the **IRMOS Control Loops** applied to the Reservation process refers to:

- *Environment Control*: The Technical-SLA reservations are communicated by the IaaS providers to the PaaS providers. Given that the Workflow Enactor Service reside on the PaaS, while an instance of it is also deployed in the VSNs, any change to the configuration from the Application Control Loop results in changes in the reservation through the Workflow Enactor Instance.
- *Virtualization Control*: The physical resources are being reserved and allocated according to the VSND, which is part of the Technical-SLA. The VSND contains functional requirements for the deployment with respect to computing, storage and networking. When the VSN needs to be instantiated the respective resources are brought into service via the Virtualization Control interface. In order to set up or configuring resources, for computing and storage the Resource Manager instructs the Execution Environment (EE) and for networking the Path Manager instructs the ISONI eXchange Box (IXB) [7].

4.3. Real-time

The goal of the channels / processes described in this section is to provide the necessary functionality to guarantee QoS during execution. Therefore, we describe (besides execution) two main processes: Monitoring and Re-negotiation. The first one is a fundamental process that allows for evaluation of metrics during runtime in order to ensure that the reserved resources meet the application requirements, while the second one may either be triggered by the application at runtime (e.g. more users in a collaborative session) or by the IaaS providers if the initially expressed application

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

requirements cannot be fulfilled with the reserved resources and re-negotiation is needed in order to guarantee the QoS.

4.3.1. Execution and Monitoring

The **goal** of the Execution and Monitoring process is to enable execution of the application according to the QoS requirements, while monitor both the application and the infrastructure to ensure that execution is performed according to the agreed QoS terms (as included in the SLAs).

The **prerequisites** for the Execution and Monitoring process are the deployed VSNs, the uploaded binaries, the workflow description document produced by IRMOS Papyrus as well as the deployed Monitoring and Workflow Enactor Instances in the VSN.

The main **actors** in this process are the *SaaS Provider*, the *PaaS Provider* and the *IaaS Provider*.

The **tools / services** engaged in this process are the following:

1. *Deployment Manager*: During the execution phase, inside the IaaS provider, the Deployment Manager is responsible for collecting VSN individual monitoring information. It collects all VSN individual monitoring reports and sends the low level information to the Monitoring Service (inside the PaaS provider) in the configured form requested during the reservation (Technical-SLA).
2. *Resource Manager*: During the execution phase the Resource Manager is responsible for collating and forwarding monitoring information received from the infrastructure regarding the resources.
3. *Path Manager*: During the execution phase the Path Manager is responsible for collating and forwarding monitoring information received from the infrastructure regarding the network links.
4. *Workflow Enactor Service*: A service, inside the PaaS provider, responsible for configuring, starting and stopping the applications (an instance of the service is deployed in the VSN to invoke the services).
5. *Monitoring Service*: A service, inside the PaaS provider, responsible for collecting the high as well as the low level information provided from the ASCs and the ISONI respectively (an instance of the service is deployed in the VSN to monitor the ASCs and provide corresponding reports to the monitoring service).
6. *Storage Manager*: During execution phase the Storage Manager is responsible for collating and forwarding monitoring information received from the infrastructure regarding the storage units.
7. *IRMOS Real-time Scheduler*: A service allowing for temporal isolation among concurrently running Virtual Machine Units, in such a way that the temporal interferences among them do not disrupt the QoS guarantees required by the applications running within the VMUs [8]. This mechanism provides strong scheduling guarantees for an entire VMU, since it ensures a configurable CPU time within a guaranteed repeating maximum period of time. The application component running inside a VMU benefits from these real-time guarantees by experiencing a constant CPU performance as if it were running alone on the physical system. Inside the VMU, in order to control how the VMU execution time

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

(guaranteed by the outer real-time scheduler) is distributed among competing processes, it is possible to exploit both real-time priorities of the guest Operating System, and to reuse the IRMOS real-time scheduler (with higher time granularity values), which is also capable of running inside a VMU. The advantage of the IRMOS scheduler over the priority-based one is the ability to provide temporal encapsulation among competing processes, ensuring that an individual process inside the VMU runs with proper QoS guarantees.

8. *ISONI eXchange Box*: A service regulating concurrent deployments regarding networking resources in order to manage and guarantee the bandwidth. Flow control ensures that the VSNs are really isolated and do not impact each other. Connectivity to outside of a VSN may be allowed, if configured beforehand e.g. by specifying public IP addresses.
9. *Storage QoS Manager*: Based on storage pool's and associated VMU connections' QoS parameters the Storage QoS Manager enforces storage quality of service on each VMU connections.
10. *Execution Environment*: The Execution Environment is a framework in which a Virtual Machine Unit is running. It also adds additional features including real-time enabled execution through the IRMOS Real-time Scheduler and features for redundancy, migration and the connection to the long term storage. It also provides an endpoint for the connection to the virtualized network for the interaction between different Virtual Machine Units.

The **flow of events** in this process is the following:

- The T-SLA Manager requests for configuration information from the SLA Negotiator.
- The SLA Negotiator submits the configuration information to the Workflow Enactor instance that has been deployed in the IaaS.
- The Resource Manager instructs the Execution Environment on the reserved physical hosts to adjust the VMU real-time configuration according to the computing requirements specified in the VSND of a Technical-SLA. In case there is a requirement for migration, this introduces the Execution Environment and the Path Manager to prepare and do the migration.
- The Execution Environment starts the VMU according to requested parameters and configures the IRMOS real-time scheduler on behalf of the Resource Manager.
- The Path Manager instructs the IXBs (according to the networking requirements specified in the VSND of a T-SLA) on the physical hosts to adjust the flow control of the related Virtual Links (VLs). On the domain level ISONI paths are selected, which are capable of satisfying required QoS attributes.
- The Storage QoS Manager is used by the Storage Manager to setup the low level parameters for the Storage QoS. The Storage QoS Manager uses the Storage Specific Technical-SLA inputs from the Storage Manager, profiles the storage subsystem through a Profiler and makes a decision as to whether the SLA can be reliably granted.
- The Workflow Enactor Service starts and stops services according to the workflow description document. The enactor is aware of temporal constraints for activities within the workflow in addition to current control and dataflow constructs.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

- The Monitoring Service gathers information both about low-level performance parameters coming from the IaaS infrastructure as well as information about high-level performance parameters coming from the ASC Monitoring script. The capabilities of the Monitoring Service are extended into aggregating and storing the information to the Historical Database (for offline usage - providing input to the Service Engineering process) and detecting on-the-fly any Application-SLA violations (online usage - providing input to the A-SLA Manager).

The functionality of the **IRMOS Control Loops** applied to the Execution process refers to:

- *Application Control*: The information provided by the application monitoring allows for application QoS provision. One prerequisite of any IRMOS enabled application is to provide self-monitoring capabilities for the QoS metrics that the application developer intends to include in an A-SLA. This mainly involves critical ASC outputs (e.g. fps of a teleconference, response time of a server etc.). This runtime information is then relayed to the Monitoring Service in order to be utilized by the IRMOS PaaS provider for taking corrective actions that will ensure that the real-time guarantees are kept throughout application execution.
- *Environment Control*: The Application-SLA and Technical-SLA metrics are being monitored during the execution since monitoring information is collected both from the applications and from the infrastructure. Violations are communicated to the SaaS and IaaS provider in order to take corrective actions. This may result to SLA Re-negotiation (as explained in Section 4.3.2).
- *Virtualization Control*: The physical resources are monitored during execution. Any violation that cannot be handled within the IaaS domain (e.g. through live migration) is reported to the ISONI SLA Manager and escalated as a T-SLA violation.

4.3.2. Re-negotiation

The **goal** of the Re-negotiation process is to provide updated resources at runtime following either a request from a Customer or the monitoring information (obtained during execution) that shows that the initially expressed application requirements cannot be fulfilled with the reserved resources and re-negotiation is needed in order to guarantee the QoS. The changes affect the VSN and refer to:

- Computational power
- Memory (RAM)
- Bandwidth
- Storage
- Lifetime of VSN.

We do not list any **prerequisites** for this process since Re-negotiation may be triggered during execution, which actually means that all artefacts and components are in place.

As in the Negotiation process, the **actors** in this process are: the *Customer*, the *SaaS Provider*, the *PaaS Provider* and the *IaaS Provider*.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

The **tools / services** engaged in this process are the following (not explained in detail since they are the same as in the Negotiation Process – Section 4.2.2):

1. *IRMOS Portal*
2. *SLA Negotiator*
3. *A-SLA Manager*
4. *Performance Estimation Service*
5. *Discovery Service*
6. *T-SLA Manager*
7. *ISONI SLA Manager*
8. *Deployment Manager.*

The **flow of events** and the **outcomes** of this process are the following:

- The Customer may change the application parameters (e.g. increased number of end users). Through the IRMOS Portal, the parameters are changed and since the application is being executed, a Re-negotiation process is being triggered by the IRMOS Portal. The process that follows is the same as the normal Negotiation process. The Application-SLA is updated with the new parameters values and the SLA-Negotiator triggers PES in order to produce an updated VSND that is communicated to the IaaS provider to update the Technical-SLA based on the updated VSND. It has to be noted that the execution of the application is not stopped but the updated VSN provides additional resources that are used by the application.
- According to the Monitoring information being collected, the SLA Negotiator may request for a re-negotiation. In this case, the PES is triggered to produce an updated VSND that will be used for the re-negotiation. The ISONI Deployment Manger starts IaaS internally a re-negotiation transaction towards the different IaaS resource management components. The request contains another Technical-SLA with the updated VSND including the new parameters. Like during the negotiation phase only the combined acknowledgment of all resource managing components result in a successful selection. After a successful resource lookup the ISONI SLA-Manager provides a Technical-SLA re-negotiation offer which can be accepted or denied.
- The CPU resource allocation for the affected VMUs may be adapted to the new QoS requirements of the application by changing on-the-fly the scheduling parameters configured within the IRMOS Real-Time scheduler (while the VMUs are running). This is possible only if enough spare resources are available on the physical hosts where the VMUs are deployed. Otherwise, live-migration of one or more VMUs to other physical hosts may be necessary. Live-migrating a VMU is a process that may be engaged while the VMU is running without any perception of QoS degradation by the Customer for all the duration of the process, except for a short period of outage of the VMU services (down-time), which may be minimized by proper techniques [9]. In those cases in which such an outage period is deemed as not acceptable due to the criticality of the timing constraints of the application, re-negotiation of the Technical-SLA will be rejected.
- The network resource allocation for related Virtual Links (VLs) may be adjusted to the new requirements. Increasing the bandwidth just can be fulfilled, if enough bandwidth is available. Changing of delay and jitter requirements are refused, since otherwise it may cause a complete new deployment on ISONI. Delay and

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

Jitter influences the location selection and chosen transport networks heavily. If a running VMU needs to be moved to another PH host inside a node, the network resources will be adapted. Moving one VMU to another node is possible, but not wished in re-negotiation case.

- As in the case of CPU resource allocation it may be possible subject to available resource constraints to modify some aspects of the reserved storage through a re-negotiation at run-time. Storage pool capacity and performance of connections to that pool may be modified as can the number of connections (changing number of VMU's). If insufficient resources are found then a rejection of the T-SLA will occur. It may be possible to achieve a reduction in the reserved resources through this process.

The functionality of the **IRMOS Control Loops** applied to the Re-negotiation process refers to:

- *Application Control*: The application configuration is changed at runtime which is reflected to the configuration information that is passed to the Environment Control Loop through the Workflow Enactor Service.
- *Environment Control*: The updated VSND that is being produced by the PES includes the updated resource estimates. The latter allows for real-time handling of QoS exceptions and changes in uncertainty throughout the application execution.
- *Virtualization Control*: The updated resource allocations are based on the new VSND as part of an updated Technical-SLA. The Deployment Manager changes the VSN according to the new VSND.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

5. Conclusions

The IRMOS Project is designing and implementing a rich set of services to efficiently operate, manage and reconfigure computing, storage and network resources under real-time conditions, providing to end users and to the associated applications the appropriate and required level of QoS. All QoS terms are dynamically negotiated and represented in SLAs between various actors in the value chain considering both application and resource level QoS guarantees. All Platform and Infrastructure capabilities are offered as on-demand services, although the architecture of the media applications varies from traditional n-tier enterprise applications to service-oriented workflows. Another important aspect of the platform is that the services and their orchestration are built and developed in a way that preserves the real-time attributes throughout the whole infrastructure including the resources, the virtual execution environments as well as networks to the applications and to the end user.

In this paper we described how real-time aspects are addressed across all layers of cloud environments as mapped and discussed based on the corresponding layers of the IRMOS architecture. The proposed approach and the architecture is being verified through technical implementations and validated with three (3) different real-time interactive multimedia applications, namely Digital Film Postproduction, Interactive Real-time eLearning and Virtual and Augmented Reality.

IRMOS	IRMOS_WhitePaper_v1.0.doc
Interactive Real-time Multimedia Applications on Service Oriented Infrastructures	Created on 27/02/2010
Whitepaper: Achieving Real-time on Service Oriented Infrastructures for Interactive Multimedia	

6. References

- [1] IRMOS Project Website, <http://www.irmosproject.eu>.
- [2] The NIST Definition of Cloud Computing, Peter Mell and Tim Grance, Version 15, <http://csrc.nist.gov/groups/SNS/cloud-computing>, 2009
- [3] IRMOS Whitepaper, "A Blueprint for Creating Interactive Applications for Virtualized Real-time Service Infrastructures", 2010.
- [4] IRMOS Whitepaper, "Intelligent Service Oriented Network Infrastructure Whitepaper", 2009.
- [5] IRMOS Project Deliverable D5.1.1, "Models of Real-time Applications on Service Oriented Infrastructures", 2009
- [6] IRMOS Project Deliverable D3.1.3, "IRMOS Overall Architecture", 2010
- [7] IRMOS Project Deliverable D7.2.1, "Initial Version of Path Manager Architecture", 2009
- [8] IRMOS Project Deliverable D6.4.2, "Final version of Real-time architecture of EE", 2010
- [9] Fabio Checconi, Tommaso Cucinotta, Manuel Stein "Real-Time Issues in Live Migration of Virtual Machines," 4th Workshop on Virtualization and High-Performance Cloud Computing (VHPC), The Netherlands 2009.