



*Empowering the Service Economy with
SLA-aware Infrastructures*



Project no. FP7- 216556

Instrument: Integrated Project (IP)

Objective ICT-2007.1.2: Service and Software Architectures, Infrastructures and Engineering

Deliverable D.B3a

Use Case Specification - ERP Hosting

Keywords:

Service Level Agreement, ERP Hosting, SaaS

Due date of deliverable: June 8, 2009

Actual submission to EC date: June 29, 2009

Resubmission to EC date: October 31, 2009

Start date of project: 1st June 2008

Duration: 36 months

Lead contractor for this deliverable: SAP AG

Revision: V 1.3 (Oct 29, 2009)

Project co-funded by the European Commission within the Seventh Framework Program (2007-2013)		
Dissemination level		
PU	Public	Yes

Document Status	
Deliverable Lead	Hui Li (SAP)
Reviewer 1	Jens Happe (FZI)
Reviewer 2	Alfonso Castro, Beatriz Fuentes (TID)
PMT Reviewer	Ramin Yahyapour (UDO)
Complete version submitted to reviewers	1 st May, 2009
Comments of reviewer 1 received	15 th May, 2009
Comments of reviewer 2 received	20 th May, 2009
Revised deliverable submitted to PMT	25 th May 2009
PMT Approval	1 st June 2009
Resubmission Reviewer	Alfonso Castro (TID)
Comments of reviewer received	26 October 2009

Contributors	
Partner	Contributors
SAP AG	Hui Li, Tariq Ellahi, Wolfgang Theilmann, Ulrich Winkler, Sylvia Scheu

Notices
<p>The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The above referenced consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law. Copyright 2009 by the SLA@SOI consortium.</p>
<p>* Other names and brands may be claimed as the property of others.</p>



This work is licensed under a [Creative Commons Attribution 3.0 License](https://creativecommons.org/licenses/by/3.0/).

Document History			
Version	Date	Author	Changes
V 1.0	2009-05-29	SAP	Reviewer 1 & 2 comments included, ready for PMT
V 1.1	2009-06-08	SAP	PMT comments included
V 1.2	2009-06-25	SAP	b-line harmonization
V1.3	2009-10-29	SAP	Revised wrt reviewer's comments

Executive Summary

Service-Oriented Architecture (SOA) represents an architectural shift for building business applications based on loosely-coupled services. In a multi-layered SOA environment the exact conditions under which services are to be delivered can be formally specified by Service Level Agreements (SLAs). However, typical SLAs are just specified at the top-level and do not allow service providers to manage their IT stack accordingly as they have no insight on how top-level SLAs translate to metrics or parameters at the various layers of the IT stack. The concept of the project SLA@SOI is a systematic grounding of SLAs from the business level down to the physical infrastructure.

The Enterprise Resource Planning (ERP) hosting use case is presented by SAP. ERP systems are heavily used in enterprises and public organizations for managing their business processes in an efficient, effective and transparent way. ERP systems belong to the most complex existing software systems as they cover huge functionalities that can be flexibly combined in a variety of scenarios. Furthermore, they must support very large, complex and possibly distributed business processes, both within (possibly global) organizations but also between different organizations. Traditional ERP systems tend to be very large, rather monolithic and hard to set up and manage. For overcoming these drawbacks, the paradigm of service-orientation has been successfully applied by SAP for building a new ERP solution that allows for simple and flexible composition and configuration of business functionality for creating customer-specific solutions in a highly efficient way. Another recent trend in this area is the on-demand delivery of software-based solutions which is particularly relevant for small and medium-sized customers who cannot afford for lengthy time and cost consuming solution implementation projects. Both of these aspects, the service-oriented architecture and the on-demand delivery model, give a strong motivation for investigating a rigid approach to multi-layered SLAs and SLA management for future SAP solutions.

This document presents a detailed specification of the ERP hosting use case from SAP. Our goal is to demonstrate holistic SLA planning and management capabilities in typical SAP landscapes. To achieve this goal an overall service manager architecture is defined, including core SLA-related features, and an initial system landscape of the demonstrator is described. Based on these three main scenarios are elaborated: firstly in the negotiation and planning phase, it is shown how the customers interact with SAP service providers to establish business SLAs. Based on business SLAs, SAP system sizing and planning can be optimized accordingly and such SLA requirements can be translated into landscape configuration parameters in application and infrastructure layers. Secondly, SAP systems are provisioned and deployed according to optimized parameters. Thirdly, at operation phase the SLAs at different layers are monitored, and proper change management procedures are triggered based on alerting or predictive monitoring. Through these three scenarios a full life cycle of service management with SLA capabilities are demonstrated for hosted SAP solutions.

Table of Contents

1	Introduction.....	9
1.1	Project Background.....	9
1.2	The SAP Use Case.....	10
2	Background.....	11
2.1	SAP Business Solutions.....	11
2.1.1	SAP Business Suite.....	11
2.1.2	SAP Solutions for SMEs.....	12
2.2	SAP IT Foundations.....	13
2.2.1	SAP NetWeaver.....	13
2.3	SAP Standard Application Benchmarks.....	15
2.3.1	Sales & Distribution Application (SD).....	16
2.3.2	Structure of SD Application.....	17
2.4	Quality Management at SAP.....	18
2.4.1	SAP Solution Lifecycle.....	18
2.4.2	Quality Management for SAP Systems.....	19
2.4.3	Analysis.....	20
2.5	SLA Management & SAP NetWeaver.....	21
2.6	Business Impact & Problem Statement.....	22
3	Architecture and System Design.....	24
3.1	Service Manager Architecture.....	24
3.1.1	Infrastructure Management.....	25
3.1.2	Service Deployment & Configuration Management.....	25
3.1.3	Service Monitoring.....	25
3.1.4	SLA Violation Analysis.....	26
3.1.5	SLA Planning.....	26
3.1.6	Process Execution Runtime.....	26
3.1.7	Process Composition Tools.....	26
3.1.8	Visualization Tools.....	26
3.1.9	Service Knowledge Base.....	27
3.1.10	Portal.....	27
3.2	Demonstrator Landscape.....	27
3.2.1	System Landscape Overview.....	27
3.2.2	System Design Considerations.....	28
4	Scenarios.....	30
4.1	SLA Driven Sizing & Planning.....	30
4.1.1	Storyline.....	31
4.1.2	Scenario Diagrams.....	33
4.2	SLA Driven Landscape Configuration.....	35
4.2.1	Storyline.....	35
4.2.2	Infrastructure Provisioning & Configuration.....	36
4.2.3	SAP Landscape Configuration.....	37
4.2.4	Capacity Validation Benchmarking.....	40
4.3	SLA Driven Landscape Management.....	41
4.3.1	Monitoring and Analysis.....	41
4.3.2	Change Management.....	42
4.3.3	Adaptation Actions.....	44
4.3.4	Scenario Diagrams.....	45
5	Conclusions and Future Work.....	47
6	References.....	48
Appendix A:	Glossary.....	49

Appendix B: Abbreviations 51

1 Introduction

This document presents an industrial use case for applying innovative SLA management capabilities as they are developed in the EU FP7 project SLA@SOI. The use case is taken from the field of ERP (Enterprise Resource Planning) applications as they are provided by SAP. More precisely, the use case analyses the application of an SLA management framework for the management of hosted ERP solutions which can be delivered to customers in an on-demand manner.

Overall, the use cases shall demonstrate business value in three main areas:

1. Dynamic Service Provisioning: Customers benefit from simplified and flexed consumption of dependable services.
2. Efficiency: Service providers achieve a reduction of TCO (total cost of ownership) in terms of energy, hardware and operational costs.
3. Service Transparency: Service providers benefit from more integrated services management which supports governance and provider agility.

Section 1.1 gives a general introduction to the background of the project SLA@SOI while Section 1.2 introduces the actual use case in more detail.

1.1 Project Background

The current industrial context of the project is one in which Information and Communication Technology (ICT) has been analyzed as an essential driving force for innovation and a core enabler of economic growth in the coming years. Within this context, citizens and industries alike are facing a fundamental *paradigm shift*, from a world where software is purchased towards a context in which *services* are selected.

There are many definitions of *service* used in different contexts. However, all are based on the same principle: a service consumer *does not own* the service and therefore need not be concerned with all the aspects generally associated with ownership such as infrastructure, technology, integration and maintenance. Instead he/she has only to choose a service which meets his/her business needs. This shift is helping technology fulfil its role as an *enabler of change* rather than as an inhibitor.

The on-demand provisioning and delivery of services according to well defined service level agreements (SLAs) is a major challenge [LTH09]:

- Software providers face the challenge of predicting the behaviour of services and service compositions with reasonable accuracy. The advent of service-oriented architectures has realized an unprecedented flexibility which comes at the cost of increased internal system complexity.
- Infrastructure providers face the challenge to dynamically allocate resources so that contractual obligations according to SLAs are met with minimal effort and costs.
- Service providers are challenged to negotiate and manage SLAs in a consistent, transparent and automated way both at business and technical level.

- Service customers face the challenge to describe and negotiate required service characteristics in a way that is meaningful to their level of expertise but also sufficiently accurate.

The overarching challenge is the need for a service-oriented infrastructure (SOI) that supports consistent SLA management across all layers of an IT stack and across the various stakeholder perspectives. Noteworthy, the SLA characteristics spans across multiple non-functional domains such as security, performance, availability, and reliability.

The concept of this project is a systematic grounding of SLAs from the business level down to the physical infrastructure. This way, SLAs can be ultimately enforced or at least automatically adjusted at the lowest possible level, namely the physical infrastructure, and can also be managed along complete service value chains involving various different providers. Noteworthy, the degree to which full SLA enforcement, adjustment or just alerting is possible depends on the specific nature of the respective SLA properties.

1.2 *The SAP Use Case*

The Enterprise Resource Planning (ERP) hosting use case is presented by SAP. Traditional ERP systems tend to be very large, rather monolithic and hard to set up and manage. For overcoming these drawbacks, the paradigm of service-orientation has been successfully applied by SAP for building a new ERP solution that allows for simple and flexible composition and configuration of business functionality for creating customer-specific solutions in a highly efficient way.

While large enterprises are already heavily using ERP systems, the uptake of such systems for SMEs is still low due to the complexity of the system setup and maintenance. Overcoming this barrier requires two main activities. First, an approach that allows for simple composition of ERP functionality. This is largely achieved by the SOA paradigm. Second, an approach that allows for simple consumption of ERP functionality. A prominent approach in this area can be found in the hosting and software as a service field which relieves SME customers from the burden of managing the IT systems on their own. However, the main obstacle for applying these approaches in the ERP context is the automated management of the provided system properties, usually specified by an SLA.

This document gives a detailed specification of the SAP use case. Our goal is to demonstrate holistic SLA planning and management capabilities for hosted SAP solutions. The rest of the document is organized as follows: Section 2 gives a general overview of SAP and its technology stack, and introduces a typical application called Sales and Distribution. The section also provides a concluding business value and problem statement based on an analysis of quality and SLA management related aspects in SAP. Section 3 gives a high-level description of the service manager architecture, including the SLA-related features. Furthermore, it describes the system landscape for the demonstrator. Section 4 presents the three main scenarios of the use case with detailed steps and diagrams. Section 5 summarizes the specification of this use case and discusses the future work.

2 Background

The objective of this section is (1) to give a business-driven overview of SAP ERP solutions, hosted ERP solutions in particular, (2) a technology-driven insight on the SAP IT stack, (3) to introduce the *Sales and Distribution Benchmark* As important reference application for our demonstrator, and (4) a detailed analysis of quality and SLA-management related activities and challenges in the context of SAP solutions. Section 2.1 details the business perspective, Section 2.2 details the technology perspective, Section 2.3 details the reference application, Section 2.4 details the quality management analysis, and Section 2.5 the SLA management analysis. Section 2.6 provides a concluding business impact and problem statement.

Enterprise Resource Planning (ERP) solutions are used to coordinate all the enterprise wide resources, information, and activities in an effective and efficient manner required to complete business processes and to fulfill business targets. Nowadays, ERP systems usually consist of a common database and a couple of tightly integrated applications tailored to the needs of various different departments. The common shared database allows every department of an enterprise to store and retrieve business process relevant information. A requirement is that this information should be processed in a real-time, easy to access, secure, reliable and consistent manner.

ERP systems are well established in large and very-large enterprises. However, the uptake for small and medium enterprises (SMEs) is still low, due to high complexity, high initial costs of system setup and maintenance. SMEs usually lack the capacity to operate a highly-available, secure data centre. This also includes the resources needed to fulfil obligations against third parties and government organizations, for example the obligation to store important tax related data for a longer period of time, ranging from months up to several years. The task of updating, maintaining and adapting the process logic of an ERP system in order to meet new laws and regularities can also become a burden. A solution could be found in external hosting and providing ERP functionality as a Software-as-a-Service (SaaS) solution. ERP systems are mission critical due to the fact that they are tightly coupled into business processes. Long downtimes, lost or corrupted information, security breaches and unresponsive user interfaces cannot be tolerated. SLAs give the customer additional confidence in hosted ERP solutions and an ERP provider may promise compensation, if a guarantee is not met.

2.1 SAP Business Solutions

2.1.1 SAP Business Suite

The *SAP ERP* is the pivotal component of the *SAP Business Suite* for large and mid-size businesses. The *SAP Business Suite* is the world's most comprehensive family of adaptive business applications, providing best-of-breed, industry-specific functionality for enterprises. The SAP software is built for complete application integration, unlimited scalability, and streamlined collaboration over the Internet.

Individually, *SAP Business Suite* applications enable enterprises to manage most critical business processes. Collectively, they form a tightly integrated business application suite that adds value to every facet of an organization's value chain.

SAP Business Suite applications are based on the *SAP NetWeaver* platform – and integration and application platform. This reduces total cost of ownership across the entire IT landscape and supports the evolution of *SAP Business Suite* applications to a services-based architecture. The *SAP Business Suite* may contain the following modules:

- *SAP ERP*
- *SAP Customer Relationship Management*
- *SAP Product Lifecycle Management (PLM)*
- *SAP Supply Chain Management (SCM)*
- *SAP Supplier Relationship Management (SRM)*

2.1.2 *SAP Solutions for SMEs*

SAP offers various solutions for small businesses and midsize companies (SMEs), designed and tailored to best fit unique needs and demands, to optimize every aspect of operation, to maximize revenue and minimize costs, to fully utilizes resources and leverage potentials.

- *SAP Business All-in-One*
- *SAP Business One*
- *SAP Business ByDesign*

SAP Business All-in-One

SAP Business All-in-One is a pre-configured ERP suite for mid-size businesses. *SAP All-in-One* offers the advantages of a full-fledged system, compatible with *SAP Business Suite* and tailored to the needs of a wide range of different areas of the business.

SAP Business One

Today's marketplace is more competitive than ever for SMEs trying to stay on top of day-to-day operations. Most SMEs are using various heterogeneous software products with limited ways to integrate them, with no interfaces between programs, or ways to exchange data or access all critical functions from a centralized solution. *SAP Business One* overcomes this problem by providing an affordable, integrated, easy-to-use business management solution designed specifically for small and mid-size businesses. It enables SMEs to manage critical business functions across sales, distribution, and financials, all in a single integrated system. *SAP Business One* takes advantage of the ubiquitous Microsoft Windows Environment, and offers interfaces to Microsoft Office products, e.g. a Microsoft Excel Interface. Unlike *SAP Business All-in-One*, *Business One* is not based on the *SAP Business Suite*.

SAP Business ByDesign

Business ByDesign (ByD) is the complete, on-demand, adaptable and comprehensive ERP solution for small and mid-size companies by SAP. ByD provides end-to-end integration for all key business areas – such as marketing, human resources management, and procurement. ByD was initially announced on 19th September 2007 and has been available since the beginning of 2008 for

customers in 15 selected countries. ByD is a hosted-only ERP solution. Thus customers are relieved from the financial and work intensive burden of customizing their specific solution or from buying and maintaining dedicated hardware. ByD allows for modular functionality deployment - the so-called scoping process. Every change made within the scoping and parameter selection phase produce an inherent consistent solution and therefore does not necessarily require the support of consultants or specialized experts. Conversely, ByD does not offer the degree of freedom of configurability, adaptability and the capability to be tailored to the very specific customer needs as the SAP Business suite does.

2.2 *SAP IT Foundations*

The objective of this section is to give an overview of the various layers of the SAP stack. After a brief description of the SAP stack, we would explain various methodologies from the SLA@SOI stack.

As mentioned previously, SLA@SOI aims to operate across the entire IT stack. SAP's IT stack is comprised of the following layers:

- At the bottom is the compute infrastructure providing the execution apparatus for the SAP's software and services layers. SAP software stack is capable of running on a large variety of processor architectures and platforms. Moreover, the software stack is able to run on the virtualization platforms as well as physical hardware.
- The software part of the stack is SAP NetWeaver technology Platform providing the core functionalities as well as the execution environments for services and business processes.
- The top of the SAP's IT stack comprises of the services and business processes.

The main focus of this section is the NetWeaver technology platform. Services layer is discussed at a length in the next section Enterprise SOA.

2.2.1 *SAP NetWeaver*

SAP NetWeaver [NW09] is a composition platform that allows organization to compose new business solutions quickly, while obtaining more business value from their existing IT investments. As the technical foundation for an enterprise service-oriented architecture (SOA) – a business-driven technology approach for greater flexibility and better cost efficiencies. SAP NetWeaver helps organizations evolve your IT landscape into a strategic environment that drives business change. SAP NetWeaver provides an open, flexible, and adaptable platform that addresses the challenges of today's IT infrastructures and tomorrow's IT evolution. SAP NetWeaver serves as the technology platform for the SAP stack, hence, has a complex multi-layer architecture. Figure 1 shows the architecture of the SAP NetWeaver technology platform. SAP NetWeaver includes a comprehensive set of components and tools. The following paragraphs describe the various components of the NetWeaver platform.

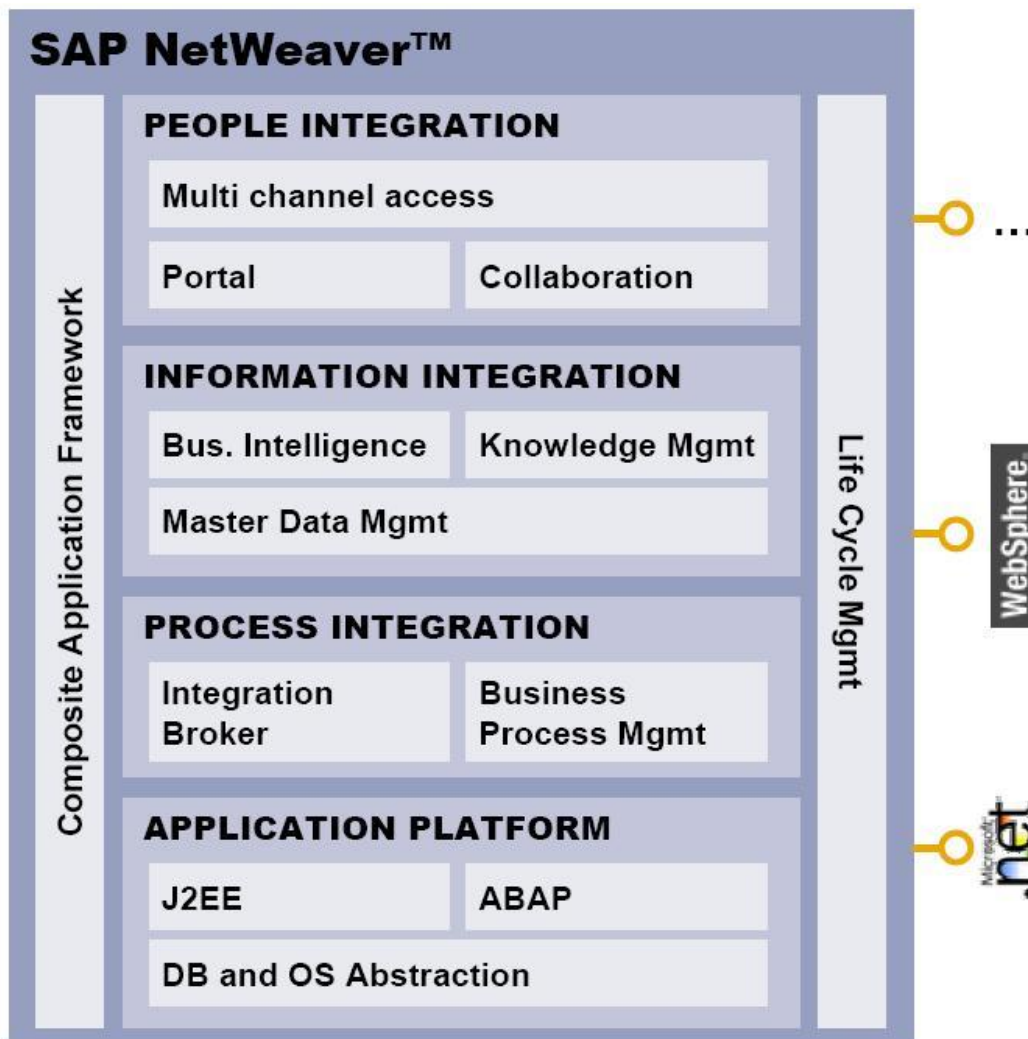


Figure 1: Architecture of SAP NetWeaver.

SAP Web Application Server is the bottom layer of the NetWeaver platform providing the execution environment for business services and processes. SAP NetWeaver Application Server brings together a proven infrastructure with the interoperability and flexibility of Web services technology. With this component of SAP NetWeaver, you get support for platform-independent Web services, business applications, and standards-based development. SAP NetWeaver Application Server provides an open and reliable infrastructure for deploying highly scalable Web applications and Web services. SAP Web AS consists of two execution environments residing side by side: Advanced Business Application (ABAP) runtime and J2EE compliant Java server. WebAS is highly scalable and reliable platform. SAP Web AS provides operating system and database abstractions empowering the business applications to portable across various operating system and database platforms.

The SAP NetWeaver Business Process Management (SAP NetWeaver BPM) component helps you model, execute, and monitor your business processes based on a common process model. It can help you improve the efficiency of business processes, reduce errors in complex repetitive tasks, and lower exception-handling costs. SAP NetWeaver BPM provides the following tools:

- Process composer – Enables process architects and developers to create and debug executable business process models. Each business process model clearly defines the rules and exceptions governing the process steps that are performed by people or systems in response to specific business events.
- Process server – Executes process models without requiring brittle translation steps between the model and code, fully integrated with the runtime technology provided by SAP NetWeaver Composition Environment.
- Process desk – Allows process users to perform their assigned tasks while business processes are running. Workers can access context-sensitive task screens through the standard SAP user interface or through universal work lists displayed via SAP NetWeaver Portal.

SAP NetWeaver Process Integration (SAP NetWeaver PI) enables organizations to manage end-to-end process integration using a standards-based, service-oriented architecture (SOA) infrastructure.

SAP NetWeaver Business Intelligence (SAP NetWeaver BI) paints a complete picture of your business to satisfy the diverse needs of end users, IT professionals, and senior management. It brings together a powerful business intelligence infrastructure, a comprehensive set of tools, planning and simulation capabilities, and data-warehousing functionality – delivered through enterprise portal technology. SAP NetWeaver Master Data Management (SAP NetWeaver MDM) is an enabling foundation for enterprise services and business process management – providing a single version of the truth for customer, product, employee, supplier, or user-defined data objects.

SAP NetWeaver Portal unifies key information and applications to give users a single view that spans IT silos and organizational boundaries. It allows you to take full advantage of all your information resources – and maximize the return on your IT investments. And, its predefined business content accelerates implementation and reduces the cost of integrating your existing systems.

The SAP NetWeaver Mobile component is the technical foundation for enterprise mobility within SAP NetWeaver. With SAP NetWeaver Mobile, your business can mobilize quickly, taking advantage of existing IT infrastructure and extending your tried-and-true business processes within and beyond enterprise boundaries.

2.3 *SAP Standard Application Benchmarks*

SAP Standard Application Benchmarks [SAPS09] help customers and partners find the appropriate hardware configuration for their IT solutions. SAP and its hardware partners developed the SAP Standard Application Benchmarks (described in more detail below) to test the hardware and database performance of SAP applications and components. The benchmarking procedure is standardized and well defined. Originally introduced to strengthen quality assurance, the SAP Standard Application Benchmarks can also be used to test and verify scalability, concurrency, and multi-user behavior of system software components, RDBMS, and business applications.

There are different methods to analyze the performance of a system. These methods can basically be subdivided into measuring methods and modelling techniques. However, to measure, compare, and normalize the service times for different platforms all approaches need a hardware independent standard. The

following two approaches to define a platform independent performance standard with the help of benchmark programs or suites are:

- Real-world reference machine. All measurement results are compared with the results obtained on a real-world reference machine (such as the MIPS7 definition). This is a simple model, but amongst others, fast development in hardware technology and portability issues are the great disadvantages of this model.
- Theoretical reference machine. An abstract machine or a theoretical reference machine is defined by its performance characteristics alone. Because it is an abstract machine, its performance can always be adapted to the state-of-the-art. Haas & Zorn suggest an abstract machine with simulations of real processes to determine the load factors. The performance is defined by throughput and response time numbers of specific programs.

SAP opted for the second approach as a more unified analysis of SAP applications. The theoretical reference programs are called *SAP Standard Application Benchmarks*. These application benchmarks simulate user or background activity for different applications, thus creating throughput. In turn, throughput numbers define the processing power of a system or configuration. For better comparison capabilities, SAP introduced a hardware independent unit called *SAPS*. 100 SAPS are equivalent to 2,000 fully processed order line items per hour, 6,000 dialog steps (screen changes) with 2,000 postings or 2,400 SAP transactions.

The most important reasons for choosing the abstract reference machine are:

- Using an abstract reference machine allows to compare different architectures as well as different client/server configurations (2-tier, 3-tier, multi-tier). This is very important when testing scalability issues.
- Due to the rapid progress in hardware technology, a physical reference machine will be outdated very quickly.
- SAP always considers throughput in the context of business processes. This also enables the comparison of SAP software with legacy systems. Furthermore, this kind of definition takes into account that the implementation and tuning of hardware and software components on different platforms may differ.

Furthermore, this kind of definition takes into account that the implementation and tuning of hardware and software components on different platforms may differ.

For SAP ERP exist various different standard application benchmarks, one of them is the *Sales and Distribution (SD) Benchmark* described in the following section.

2.3.1 Sales & Distribution Application (SD)

A starting point for hosted SAP ERP solution is the so-called SD benchmark.

The Sales and Distribution (SD) benchmark covers a core business process including selling, shipping and delivery. The SD benchmark is an overall good basic use case for the following reasons:

- It is the most visible within the set of SAP benchmarks, and it is one of the most credible and influential high-end OLTP benchmarks in the industry.
- SD benchmark focuses on CPU utilization. Customers typically have high scalability demands for their specific SD scenarios.

- It is easy to install, use and modify.
- It is well understood and accepted; therefore many comparative studies are available.

2.3.2 Structure of SD Application

The Sales and Distribution (SD) application covers a sell-from-stock business process which includes the creation of a customer order with five line items and the corresponding delivery with subsequent goods movement and invoicing. It consists of the following six transactions:

- Create an order with five line items. (VA01)
- Create a delivery for this order. (VL01N)
- Display the customer order. (VA03)
- Change the delivery (VL02N) and post goods issue.
- List 40 orders for one sold-to party. (VA05)
- Create an invoice. (VF01)

There are fifteen dialog steps with 10-second think time in-between.

The detailed benchmark steps are shown in Figure 2.

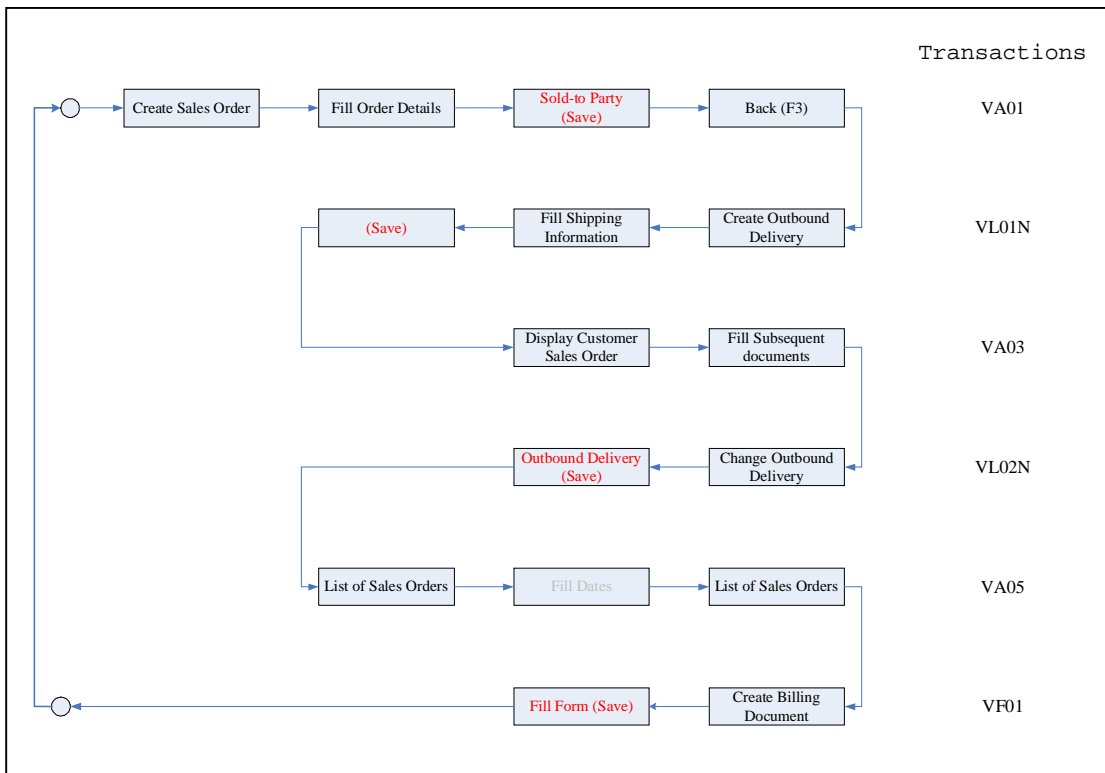


Figure 2: Regular steps within the SD benchmark.

The SD-based application serves as a basis for demonstrating SLA-driven planning and management features for hosted SAP solutions.

2.4 Quality Management at SAP

The objective of this section is to provide an industrial perspective and analysis on the relationship between architectural issues, quality concerns and requirements for a holistic SLA management in SAP products [TK08].

Quality concerns have always played an important role in the design of business applications at SAP. However, current cost pressure in IT industries has led to an even more important role of quality considerations in the development process and the design of SAP architectures. Due to the emerging need for hosted business software and on-demand ERP systems (see section 2) advanced service and service level agreement management has been identified by SAP architecture analysts as an promising approach to address service quality issues and costs related concerns.

2.4.1 SAP Solution Lifecycle

Quality concerns need to be dealt with along the complete lifecycle of a solution. The following figure addresses the main phases and steps of this lifecycle at SAP.

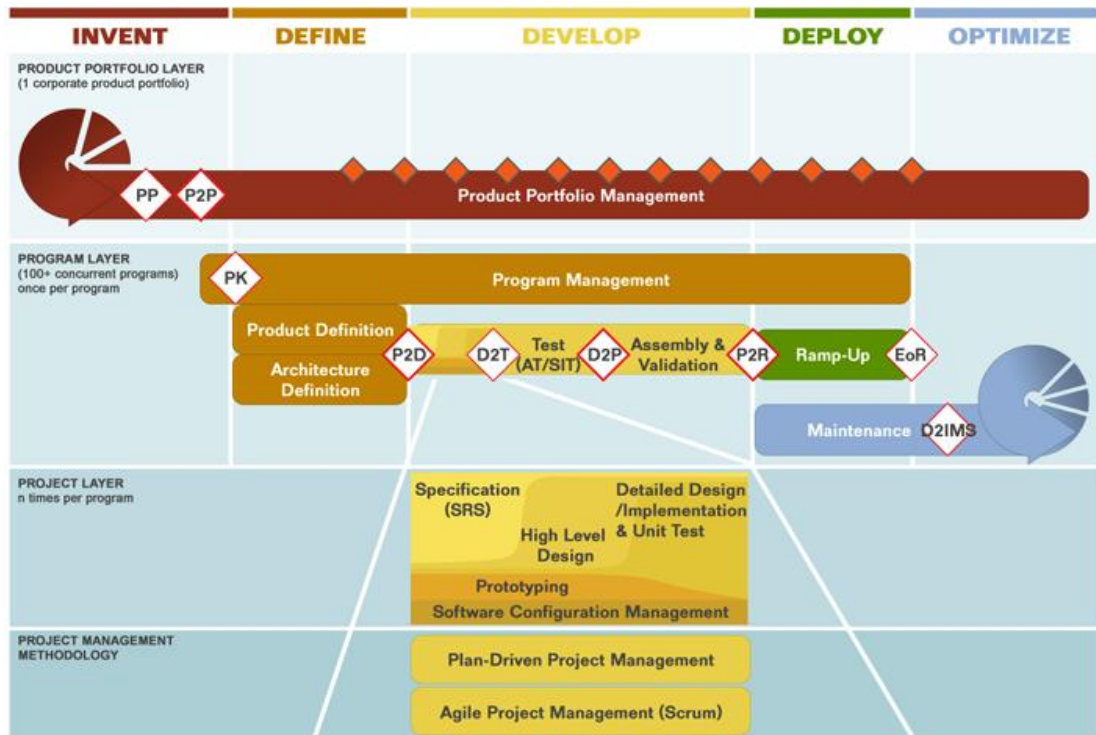


Figure 3: Product innovation lifecycle at SAP

Along this lifecycle the following main roles deal with the following issues:

1. Solution Manager provides estimated quality KPIs.
2. System architect specifies assumed characteristics of building blocks.
3. Component Designer relates requested quality with characteristics of underlying components.
4. Developer refines previously assumed quality characteristics,
5. Developer derives actual quality characteristics from unit tests.

6. Tester validates requirements from solution manager.
7. Consultant/Customer specifies actual artefacts to be used (=> first TCO estimate).
8. Consultant/Customer specifies actual landscape (=> final TCO determination).
9. Administrator observes system whether quality targets are met,
10. Consultant/Customer explores impact of planned changes.

Noteworthy, that this lifecycle is a quite simplified abstraction in 2 ways: First, the actual solution lifecycle includes various cycles and feedback loops which allow for an iterative development process that can benefit from early prototypes and tests and feed their results back into earlier stages of the lifecycle. Second, actual development almost never starts from scratch but faces an already existing system which needs to be modified or extended. This obviously limits the freedom but also the uncertainty for all the involved stakeholders.

2.4.2 Quality Management for SAP Systems

SAP systems range from midsize to very large size systems, the latter consisting of several hundred million lines of code and deployed on hundreds of distributed compute nodes. In addition to this sheer size, there are a couple of other relevant characteristics which are briefly sketched below.

Several kinds of Architectures: There is no single architecture of an SAP system that covers all perspectives of the relevant stakeholders. Specific architectures are

- The *SAP NetWeaver technology platform* (described in section 2.2.1) providing generic IT platform and integration functionality,
- the *business process platform*, providing general business logic which can be flexibly used, combined and composed,
- *applications architectures*, representing actual and possibly customer or industry specific solutions,
- *service architectures*, embedding an IT solution into a bigger business service solution, and
- *System landscapes*, describing the actual infrastructure that operates an IT solution and its configuration.

All these architectures provide a different view on different quality concerns for different stakeholders and at different granularity. A proper development process must ensure that information flows correctly between these different perspectives and overall quality concerns can be properly managed and achieved.

System (Development) Paradigms. SAP has adopted the paradigm of service-orientation for developing and providing business functionality. The so-called Enterprise SOA approach goes far beyond regular Web services as Enterprise services feature clear business semantics (they are structured according to a harmonized enterprise model based on business objects, process components, and global data types), quality and stability (they safeguard a stable interface for future versions) and adherence to standards (they are based on open standards such as WSDL and UN/CEFACT CCTS).

Furthermore, a sound model-based development approach has been chosen which provides multiple rich models for both business (integration scenarios,

process components) and IT perspective (business logic, integration logic, configuration).

Customer Engagement. Customers are involved in the specification of new SAP solutions already in the very first phases of a solution lifecycle as depicted in figure 3. This co-development serves for creating solutions which eventually meet market needs in terms of functionality but also quality requirements. However, the (quality) requirements and environment of specific customers are largely unknown at design time. This significantly increases the difficulty at design time to predict quality properties and costs of only vaguely known target environments. Consequently, traditional SAP systems require thorough go-live check at a customer site before they can become operational. Incorporating configurable and adjustable parameters for quality requirements into the solution lifecycle in means of SLA management will contribute to simplify, streamline and systemize the process of quality management of SAP products.

Cost Concerns @ SAP. The costs of IT systems are mainly determined by 3 factors: cost of engineering, cost of provisioning and cost of operation. Exploiting economies of scale significant further cost reduction can be achieved by increasing the effort in developing solutions of high quality which then allow for provisioning and operation at lower costs. Of course, this only works out if there is a positive trade-off between the additional effort and the saved provisioning/operation costs.

Quality Concerns @ SAP. Having the characteristics presented above in mind, the most important quality aspects addressed in SAP are:

- *Scalability* is clearly a cross-cutting concern that spans across all kinds of architectures. However, scalability is largely solved by the technology platform which allows for linear scalability of the application server cluster.
- *Availability* is largely achieved in the technology platform by a fault tolerant setup of application server & database.
- *Responsiveness (response time)* is again a cross-cutting concern. There is a global requirement of 1-2s maximal response times for interactive applications. This requirement is broken down via budgeting to architectural layers and components.
- *Efficiency (resource consumption)* is another cross-cutting concern. A sizing formula allows relating usage profile per applications with resource demands. There are currently running efforts to build a sizing repository (for components). However, actual customer requirements for efficiency can be very specific and SMEs in particular are very cost-sensitive. This motivates current research efforts for multi tenancy support, i.e. resource sharing between different customers/tenants.
- *Extensibility (as part of maintainability)* is mainly addressed at the business process platform. Here a dedicated framework part of the architecture assures various degrees of seamless extensibility.
- *Portability* is solved by the technology platform by avoiding usage of any hardware/OS/DB-specific features.

Other quality aspects such as usability and security are mainly addressed by development guidelines and do not directly reflect in any architecture.

2.4.3 Analysis

The following figure qualitatively summarizes the complexity of the various quality concerns at SAP in terms of architectural complexity (i.e. how much

architecture are involved) and process complexity (i.e. how many stakeholders and lifecycle phases are involved).

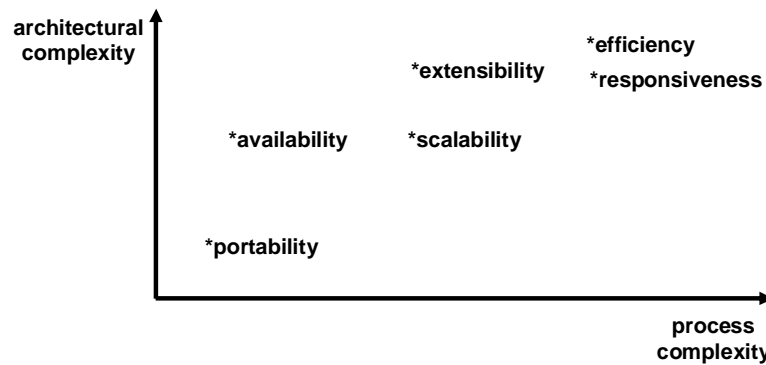


Figure 4: Architectural and process complexity of quality concerns at SAP.

As illustrating example, some specific challenges for managing efficiency are sketched out.

First, there is the underspecified environments which means that (a) concrete deployment and infrastructure (hardware, DB, OS) are unknown at design time, (b) customer requirements/behaviour unknown at design time and still underspecified at go-live time, (c) actual control flow is known vaguely (at design time) and slightly better (at testing time – scenario-based testing) again better after business configuration and even better at run-time, (d) component developers are focused on one architectural layer while non-functional characteristics of lower layers are only vaguely specified and subject to change, (e) the number of configuration & usage variants prevents from exhaustive testing and (f) scenarios of dynamic service composition are even harder to predict.

Second, the various architectures and programming models are just loosely coupled which means that no formal/provable relationship between architecture models and programming artefacts exists. It is currently unclear whether a closer coupling is feasible at all with general purpose programming languages.

Third, technical expertise on non-functional behaviour of artefacts is widely spread and poorly formalized, so it's hard from an overall perspective to say who knows/does what and when.

2.5 SLA Management & SAP NetWeaver

In addition to the discussion of quality management in the previous section, this section now serves for discussing actual SLA management issues in the context of SAP NetWeaver. As described in Section 2.2.1, SAP Web Application Server is the core component of the platform providing execution environment for the business services and processes. SLA management activities in the context of SAP stack would focus on the SAP Web Application Server. Behavior and performance of the Web AS components would impact the performance of the business services and processes. In order to control and enforce the service level objective at the services layer, the execution behaviour of the Web AS must be handled. For example, Web AS employs a number of worker processes to execute the application logic realizing the services at higher layer. The number and behaviour of these processes impact the performance and behaviour of the services.

SLA translation methodologies must be extended to translate the high level business service level objectives like response time and throughput to Web AS related metrics like number of worker processes execution application logic or sizes of various buffers etc. On the same note, SLA monitoring activities should take into account the events generated from the Web AS components and to perform analysis to identify potential SLA violation possibilities.

Similarly, it would be of vital importance to investigate the relationships between the infrastructure requirements of the logical resources in SAP Web AS and underlying compute resources. This can help service providers during the SLA planning and translation activities to identify the computational resources required providing and sustaining the QoS guarantees as stipulated by the service level agreements. Mappings of Web AS level logical resource metrics to the infrastructure metrics like number of CPU, memory etc is also vital for SLA compliance monitoring. On the other hand, relationship between infrastructure level metrics and the Web AS metrics can be utilized to predict the behaviour of Web AS based on the historical data gathered by infrastructure monitoring and this enabling proactive SLA management.

2.6 Business Impact & Problem Statement

ERP systems are heavily used in enterprises and organizations for managing their business processes. Traditional ERP systems tend to be very large, rather monolithic and hard to set up and manage. A new on-demand solution is called that allows for simple and flexible composition and configuration of business functionality for creating customer-specific solutions in a highly efficient way. The ERP hosting use case follows this vision, and addresses SLAs and SLA management topics for delivering ERP applications as a service. The envisioned research results have the potential to be reflected and included in SAP's next-generation on-demand portfolios, contributing to the competitive advantage of both functionality and non-functional properties. Such solutions also deliver potential business values by the reduction of total cost of ownerships (TCO) while providing performance and service level guarantees.

The discussion in this chapter shows a clear need to introduce SLA management driven from the business perspective. In this context, we can distinguish the following main business goals depicted in the table below.

Business Area	Business Value
Dynamic Service Provisioning	Ease of Service Consumption Dependability Flexibility
Efficiency	Energy Efficiency Technical Efficiency Operational Efficiency
Transparent Service Management	End2End manageability Service Governance Agility

Table 1: Main business areas and related business value metrics.

The first area of “Dynamic Service Provisioning” is about delivering trusted value to customers. This should become evident in aspects such as ease of service consumption (the simplification of the process for actually consuming a desired service), dependability (i.e. the trustworthiness of the non-functional aspects that complement the functional scope of a service), and flexibility (the ability for customers to change previous service agreements).

The second area of “Efficiency” is addressing service providers and provides business value in terms of reduced TCO (total cost of ownership). This boils down to environmental efficiency (here the energy efficiency of a hosted solution), hardware efficiency (the amount of hardware and its utilization) and operational efficiency (the effort to maintain a running system, respectively to complete certain management tasks and procedures).

The third area of “Transparent Service Management” constitutes a more qualitative value for service providers. It includes issues such as end2end manageability (i.e. the consistent and integrated management of services, SLAs and resources), service governance (the overarching control of multiple co-running services and service offers), and agility (i.e. the provider’s ability to change a running system in order to reflect changed business conditions).

While the expected business benefits are pretty clear, we saw at the same time the big gap between currently established practise on quality management and an envisaged automated SLA management.

This gap suggests to follow a stepwise approach in order to research for and realize the envisage target scenario. This means to start from a simplified experimental setup and then to extend it in an iterative manner. Therefore, we decided to start with experiments based on the well-understood sales and distribution benchmark application (SD). Furthermore, the initial focus on non-functional properties is set on performance-related quality attributes as analysed in Section 2.4.2.

Based on the scenarios sketched in Section 4, we demonstrate the holistic SLA management solution in SAP landscapes. Such solutions are explored in the following 2 ways: Firstly, a top-down approach is followed for analysing how business-level SLA objectives are to be translated into lower level application and technical SLA management capabilities. Secondly, a bottom-up approach is adopted for the operation management of the SLA lifecycles starting from isolated experiments which are later combined into a comprehensive solution.

3 Architecture and System Design

This section describes the architecture of the service manager and the overall system design for the ERP Hosting Demonstrator.

3.1 Service Manager Architecture

Figure 5 shows the architecture of the service manager. The individual architectural elements of the demonstrator are discussed in the following sections.

SAP ERP Hosting demonstrator will be built on the components developed by the consortium. In order to accommodate the ERP Hosting specific requirements, the existing components will be extended as well as new SAP specific components will be developed for the demonstration purposes. Figure 5 depicts this fact and shows three different types of components in the ERP Hosting Architecture diagram. It is worth noting that as part of the requirements gathering process in the project, SAP use case specific requirements have been provided and implemented in the SLA@SOI components. Considering the complexity of SAP's software stack, it is also required to have SAP-specific extensions in addition to the SLA@SOI artefacts. For instance, planning of ERP applications as hosted services leads us to develop a queueing analytic model, a TCO model, and a novel optimization technique [LCE10]. The generic aspects of this research are carried out in A6 under the task of design space exploration in which SAP is one of the main partners. Another example is monitoring, SAP software has specific sensors and monitors which are readily available for providing monitoring data, these need to be integrated with the SLA@SOI monitoring tools and rule engines. Given these facts we have designed a hybrid service manager architecture as below.

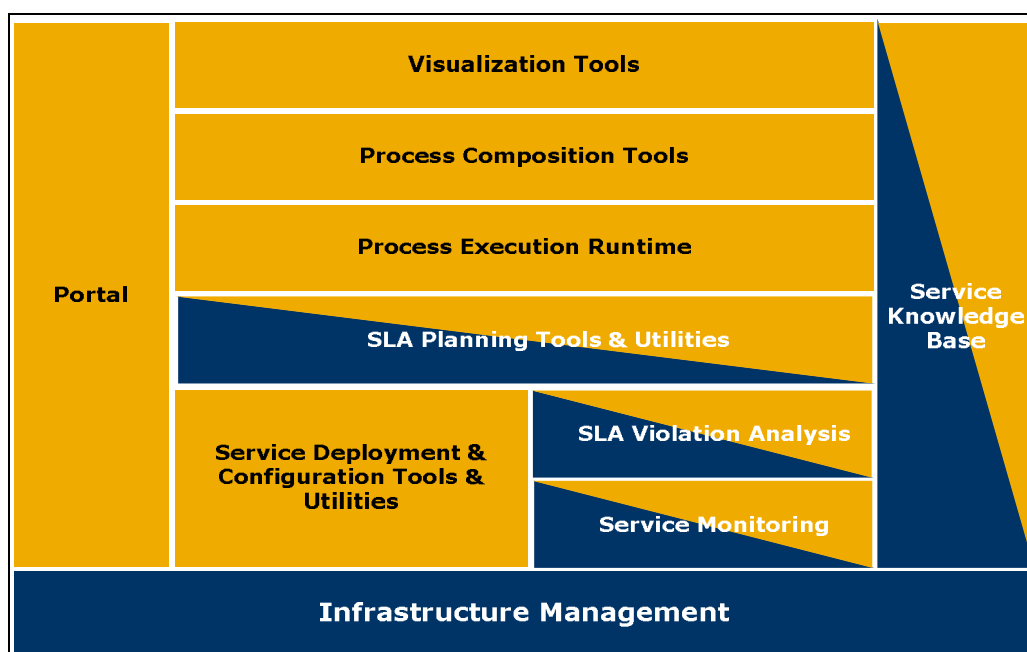




Figure 5: Service Manager Architecture.

Components colour coded in blue represent the artefacts developed by the consortium in SLA@SOI. In ERP Hosting architecture, there is only one component Infrastructure Management developed by the Work package A4. Components colour coded in Orange are developed by SAP for the use case demonstrations. Whereas, components colour coded with Blue / Orange are SLA@SOI artefacts extended with ERP Hosting specific extensions.

3.1.1 Infrastructure Management

The bottom layer in Figure 5 represents the physical / virtual IT infrastructure management apparatus. This layer would be responsible for the management of compute, network and storage elements of the execution stack. This layer will be realized via the SLA@SOI Infrastructure Management component developed in work package A4.

3.1.2 Service Deployment & Configuration Management

This layer consists of a set of tools and utilities for deployment and configuration management of SAP software stack. These tools and utilities can be composed in higher level operational processes to carry out a specific task e.g. Service Provisioning or change management etc. The focus would be to automate these tasks and to be invoked automatically but focus should be to allow manual consumption of these tools by the administrators. It would be of vital importance to investigate some uniform interface for both automated and manual consumption of these tools and utilities to allow operational evolution of this component. New tools and utilities can be developed and plugged into this component to meet the future requirements.

3.1.3 Service Monitoring

Service monitoring component is responsible for data collection and dissemination to the interested actors (other components or human administrators). The functionality of this component can be primarily divided into two categories: 1) service operational data collection i.e. collecting information about the service metric of interest and 2) information dissemination. For this aspect, either this component can adopt a publish subscribe component which can be used for metric dissemination embedded within this component, or another individual component can be introduced to the overall architecture serving as a bus to disseminate information to interested actors. The service monitoring component is also responsible for publishing / archiving the collected data to the service knowledge base components which is described in the coming section.

The component will be built using the SLA@SOI monitoring tool developed in work packages A3. Specifically we will use the rule-based engine for event correlation. This might also involve some form of instrumentation or adapters that

can convert SAP specific information to the event format designed within the consortium.

3.1.4 SLA Violation Analysis

The SLA Violation Analysis component is the complementary component to the service monitoring. This component is responsible for metric correlation and analysis for detecting possible SLA violations. The SLA violation detection analysis will leverage the information gathered by the service monitoring component as well as the historical information for identifying SLA violations. Similar to the Service Monitoring, the components for SLA violation analysis developed by work package A5 will be reused.

3.1.5 SLA Planning

Service Planning component is responsible for sizing / planning related activities. The service planning component's functionality represents the first point of processing as soon as an SLA is established. This component will carry out the SLA decomposition procedure to transform business level SLA to lower level technical SLAs. Additionally, this component will carry out the SAP landscape design and calculate the optimal set of configuration parameters required to support the SLA in processing. SAP software stack is a complex multi-tier application; hence, some extra logic would be required for SLA planning. The generic parts of capacity planning and prediction capabilities are carried out in work package A6 and will be adopted from there.

3.1.6 Process Execution Runtime

The SLA management framework for hosted ERP solution aims to adopt the process based approach. This approach is adopted to align our framework architecture with ITIL best practices and guidelines. The tools and utilities residing at lower level components like deployment & configuration, monitoring, analysis can be composed into higher level operational processes which can then be reused. The Process Execution Runtime layer is responsible for the execution and enactment of the process workflows. This layer will rely on the lower level components to execute the operational processes.

3.1.7 Process Composition Tools

This component represents the set of tools and utilities which facilitate composition, management and maintenance of the operational processes in place [CE09].

3.1.8 Visualization Tools

Visualization tools are composed of a set of tools and utilities which can facilitate visualization related tasks. For example, these tools can be used to visualize historical monitoring data, view the process execution status, visualize the SAP landscape design etc. The design of these tools should be flexible enough to be used in a standalone fashion either through the portal or offline as well as to be invoked programmatically.

3.1.9 Service Knowledge Base

The Service Knowledge Base represents the data store in which all the service execution related and management related knowledge is captured during the design and operational phases of the service lifecycle. The Service Knowledge Base does not represent a single database containing all the required information. The objective is to adopt the CMDBf (configuration management database) standard and various different data sources should be combined into one logical data store. All other components of the framework will produce / consume this data store to carry out their tasks. The Service Knowledge Base will use the software landscape from task A3.2 as well as the SLA registry from task A5.2.

3.1.10 Portal

The Portal represents the user interface component of the framework. It is the point of interaction between various types of users e.g. customer, administrators, managers etc to interact with the framework. The portal can also be used for the visualization purposes. The portal might reuse some UI elements from work package A2.

3.2 Demonstrator Landscape

3.2.1 System Landscape Overview

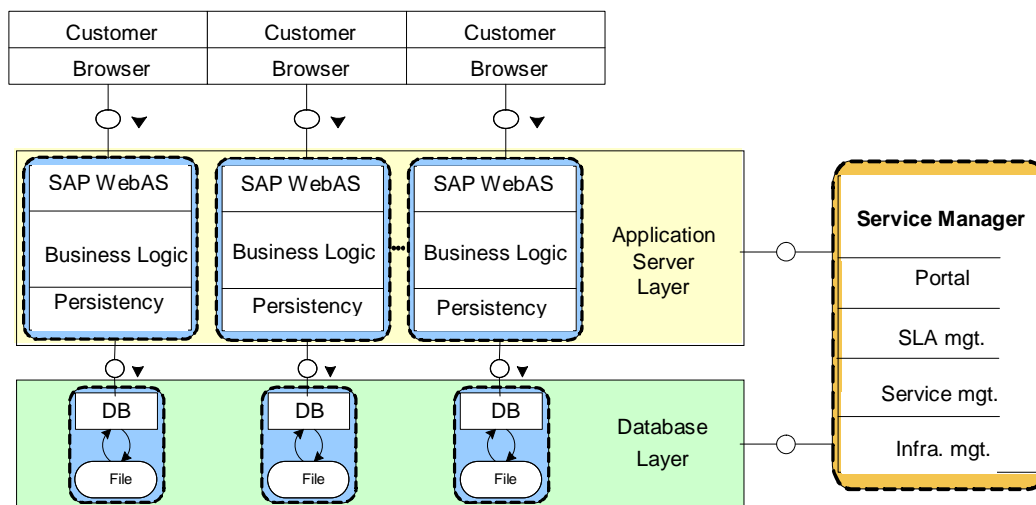


Figure 6: The system landscape for the ERP Hosting use case demonstrator

After the introduction of the service manager architecture, we describe how a service manager connects to the use case demonstrator in a system landscape. Figure 6 shows a typical setup of such a demonstrator landscape with a system-centric view. We describe the involved components in the following:

- Customer: The customer interacts with the service provider via standard web browsers, both for SLA negotiation and running business applications. The

business application used in the demonstration is the typical SD (Sales & Distribution) process described in Section 2.

- **SAP WebAS (Web Application server):** The SAP NetWeaver Application server is the central application component for business logic and persistency access control. It consists of a dual-stack, namely ABAP and Java stack. The main business logic is implemented in ABAP while UI and SOA parts are implemented in Java. SAP WebAS has its own locking and enqueue mechanisms for persistency control, and it connects to almost all major database back-ends including MSOL, Oracle, DB2, and others.
- **Database:** The selected database used in the demonstrator is MaxDB. It can be run on the same physical server as the application sever (2-tier setting), or it can be run separately on another sever (3-tier setting).
- **Service Manager:** The service manager is deployed in a separate server in the landscape of the service provider. It implements the functional building blocks and components described in the previous section. The portal is the gateway to the customer for negotiating business-level SLAs and managing customer-related matters. SLA management relates to the SLA planning, monitoring components in the service manager architecture. Service management includes service configuration, deployment, and other service utilities. Infrastructure management deals with the system-level monitoring and administration. Service manager is the central system in the landscape that manages the service lifecycle that consists of negotiation, planning, operation, and change management phases. SLA-related management capabilities are built in the service manager components and are the core features that will be demonstrated in our use case.

3.2.2 System Design Considerations

For reducing the implementation complexity and focusing on the core SLA-related features, there are several practical design considerations for the demonstration landscape:

- **One type of application:** currently we focus on one type of application, which consists of a Sales and Distribution (SD) business process. As introduced in Section 2, this is a simple yet representative process that captures the key characteristics of typical transactional business applications. And the related SD benchmark is the most-used SAP Standard Application Benchmark. We think it is a good choice from an application perspective, especially as a starting point in prototyping and Proof-Of-Concept (POC) phase.
- **Single tenancy:** Currently we will plan and provision one system per customer (single-tenant), starting with a two-tier setup. This is the simplest case and the easiest to set up. Nevertheless, this is not an optimal setup from the service provider perspective as one single hosted ERP system requires substantial initial investments. Multi-tenancy would be better option for hosted solutions. However, due to inherent software constraints these are not feasible choices for the demonstrator. For single tenancy we investigate how to optimally size a hosted ERP system given customer's workloads and SLA requirements.
- **Virtualization:** At the infrastructure layer we make use of virtualization technology for better server utilization and consolidation. This means that in a typical two-tier setting there will be one virtual machine provisioned for one customer. How to optimally set resource configuration parameters (CPU

share, threshold, etc) is one important research topic for SLA-driven planning and optimization of system landscapes.

The system landscape provides the system-related information for better understanding the demonstration scenarios, which are elaborated in the next section.

4 Scenarios

After introducing the service manager architecture and the system landscape, this section describes the SAP use case scenarios from a high-level view.

Overarching the various use case scenarios we can see the following roles involved.

Actors	Overview
Customer (Bob)	The customer of a SAP SaaS solution
SAP Sales Officer (Alice)	The business expert from the sales department who decides on the portfolio of offered SAP SaaS solutions.
SAP IT SaaS Architect (Trudy)	SaaS IT architect who supports portfolio planning from a technical perspective
SAP SaaS Administrator (Sam)	The administrator of SAP software and services (SaaS)
SAP IT Administrator (Joe)	The administrator of a SAP infrastructure SaaS provider

4.1 SLA Driven Sizing & Planning

This scenario focuses on the SAP sizing and planning aspects. The objective of the scenario would be to show that, based on the customer requirement agreed through the established SLAs, the system will facilitate in the design of the landscape that will be able to sustain the service levels. For example, planning the SAP landscape would involve identifying the number of dialog instances required as well as how these are going to place in relation to the central instance and database instance. Whether there should be a separate database instance or database and central instance should be together on the same machine. Additionally, based on some high-availability and fault tolerance requirements, some other infrastructure is required e.g. clustering etc.

The SaaS provider of hosted SAP solutions needs to define the metrics to be included in SLAs, and identify the parameters for optimizing TCO on his side. The necessary specification includes:

- *Usage pattern*: Number of users (tune-able by the customer for the first release), think time, the selection of processes
- *SLO metrics*: response time, throughput
- *Cost factors and TCO*
- *Tune-able parameters*: resource specification (CPU, memory, disk), #Tiers, #AppServers, #WorkProcesses, etc

The SaaS provider has to decompose SLO metrics into landscape configuration parameters for the given usage pattern. He/She also tries to optimize TCO on his/her side at the same time satisfying the SLO constraints.

This section will start with a storyline to show the steps for a visual demonstrator of the scenario, followed by a technical process diagram to illustrate such a scenario.

4.1.1 Storyline

The storyline is explained in detail via a list of steps which is shown on a UI screen.

Steps:

- 1) SAP Sales Officer (Alice) decides on the portfolio of offered SAP SaaS solutions. The output of this step is an SLA offer profile (see Table below) that specifies the default SLA terms such as usage patterns, user class, SLOs and cost. Alice sets up the initial values for the usage patterns and user classes in the default offer profile. Alice sends a request to the SAP SaaS Architect to start the planning phase.

SAP Sales Officer View			
Usage	Class	SLOs	Cost
250 < #User < 500	Large	Response Time < 2 sec	
#Users <= 250	Standard	Response Time < 2 sec	

- 2) SAP SaaS planner Architect (Trudy) starts the initial planning phase driven by the default SLA offer profile. Based on SLAs, Trudy tries to derive a set of configuration parameters that are able to satisfy the SLO constraints given the default usage patterns. Besides meeting the SLO constraints, Trudy tries to optimize TCO on the provider side. Such a planning phase, or so-called SLA decomposition, is essentially an offline optimization process. The outcome of this planning phase is initial cost estimations for all user classes, and the corresponding optimized parameters for hardware and landscape configuration. Trudy saves the planning results and sends the initial cost estimation to Alice.

SAP SaaS Architect View				
Usage	Class	SLOs	Cost	Config. Parameters*
250 < #User < 500	Large	Response Time < 2 sec	Initial estimation cost	Hardware specs. Landscape config.
#Users <= 250	Standard	Response Time < 2 sec	Initial estimation cost	Hardware specs. Landscape config.

- 3) SAP Sales Officer (Alice) receives the initial cost estimation from Trudy, takes into account other cost factors, and makes the final price quotes for the default offer profile.

SAP Sales Officer View			
Usage	Class	SLOs	Cost
250 < #User < 500	Large	Response Time < 2 sec	e.g. \$699 p.m. Penalty = \$30 p.m.
#Users <= 250	Standard	Response Time < 2 sec	e.g. \$499 p.m. penalty = \$20 p.m.

- 4) SAP SaaS Administrator (Sam) is informed that the offer profile is available and rolls out the SaaS offerings on the provider's web site. Sam has the following views in the system.

SAP SaaS Administrator View					
Customer No.	Customer Info.	Usage	Class	SLOs	Cost
1	Customer 1	250 < #User < 500	Large	Response Time < 2 sec	e.g. \$699 p.m. Penalty = \$30 p.m.
2	Customer 2	#Users <= 250	Standard	Response Time < 2 sec	e.g. \$499 p.m. penalty = \$20 p.m.

- 5) The customer who's interested in SAP SaaS offerings reviews the offers. He/she can either choose an existing offer from the default profile, or he/she can customize the usage patterns (i.e. #users) and SLOs and send the requests to SAP SaaS provider.

Customer View			
Usage	Class	SLOs	Cost
#Users <= 250	Standard	Response Time < 2 sec	e.g. \$499 p.m. penalty = \$20 p.m.
250 < #User < 500	Large	Response Time < 2 sec	e.g. \$699 p.m. Penalty = \$30 p.m.

- 6) SAP SaaS Administrator (Sam) receives customer requests for SD-on-demand:
- If the request is customized, go to step 7,
 - If the request is from the default profile, go to step 8.
- 7) Repeat step 2 and 3 for a customized request until an offer is generated. Sam is informed and communicates with the customer on the offer until an agreement is reached. SAP SaaS architect Trudy saves the planning results to the repository.

- 8) Sam forwards the final quote(s) to customer.
- 9) Customer accepts a quote and sends back the contract with agreed SLAs.
- 10) SAP SaaS admin saves the customer profile and initiates the landscape deployment and configuration process.
- 11) SAP IT administrator is informed and starts the deployment process with optimized configuration parameters.

SAP IT Administrator View		
Customer No.	Customer Info.	Config. Parameters*
1	Customer 1	Hardware specs. Landscape config.
2	Customer 2	Hardware specs. Landscape config.

4.1.2 Scenario Diagrams

UI screen:

As mentioned in the above steps, the scenario requires a data store for offer profile and 5 basic screens/views from different stakeholder perspectives:

- 1 for SAP sales officer
- 1 for SAP SaaS architect
- 1 for SAP SaaS administrator
- 1 for customer
- 1 for SAP IT administrator

A role-based access scheme will be implemented for web-based UI.

Diagram (next page):

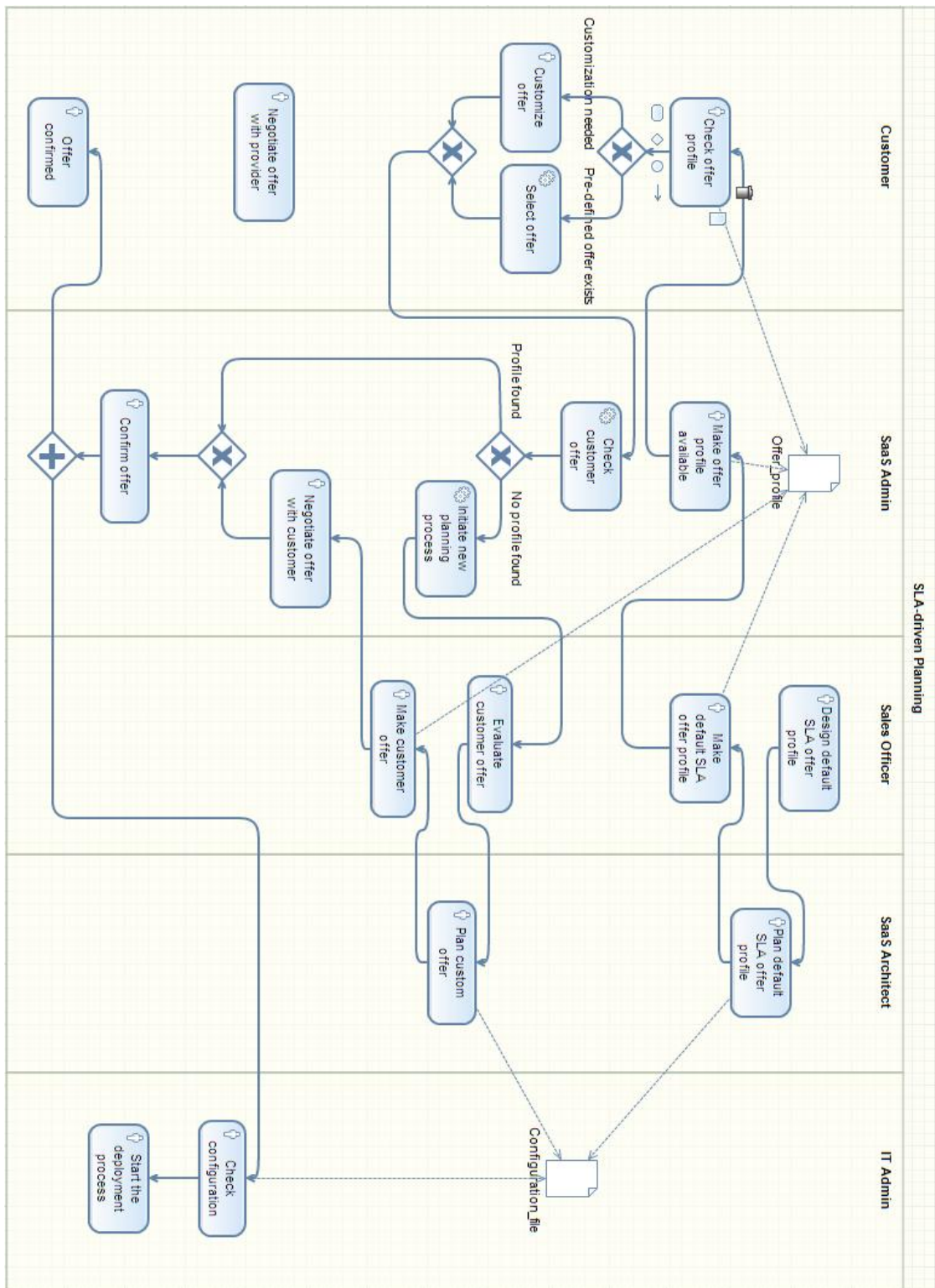


Figure 7: A process diagram, designed using SAP NetWeaver Composition Environment, illustrates the scenario steps of SLA-driven planning.

4.2 SLA Driven Landscape Configuration

The objective of this scenario would be to demonstrate the automated deployment & configuration of the SAP landscape. The configuration parameters would be derived from the SLAs as a result of the SLA-driven sizing and planning. The configuration parameters would be at both SAP stack level as well as the virtualized infrastructure level. At the level of SAP stack, the configuration parameters could be number of dialog and update work processes, whereas, at the infrastructure level, it would be the virtualized resource parameters like CPU, memory etc. Additionally, this scenario can also focus on configuring the policies at both virtualized infrastructure as well SAP stack level. The policy configuration would focus on configuring the monitoring and management thresholds and rules.

4.2.1 Storyline

The actors mentioned in the table below will be involved in the deployment & configuration phase of the service provisioning.

Steps:

- 1) SAP Service Administrator retrieves the landscape design and configuration parameters. After validation of the values, the deployment and configuration procedure is started. This could be completely automated, manual or a combination of both techniques. However, a workflow type mechanism should be in place to proceed through the different steps of the overall service provisioning process. The workflow execution can either communicate the information to appropriate personnel (a UI screen on his portal) or could feed in the information to the software component responsible for the next steps in the chain. For example, in this case the Service Administrator can validate the capacity design and submit his/her approval so that the IT administrator who is responsible for the provisioning & configuration of the virtualized infrastructure can proceed with the required steps on his behalf (mentioned in step2).
- 2) The second step requires the provisioning of virtualized infrastructure resources as calculated during the sizing / capacity planning phase. The IT administrator is responsible for carrying out these tasks. This involves the provisioning of compute (virtual machines), storage and network resources. The details of this step are presented in the next section with elaborate details and different possible mechanisms which can be utilized to carry out this task. The IT administrator retrieves the infrastructure requirement specification and configuration parameter details and triggers the infrastructure instantiation procedure. After the virtualized infrastructure has been provisioned the IT administrator must configure the virtual machines and the operating systems if applicable. The IT administrator, after the successful completion of this step signals the Service Administrator for the follow up steps. Again, this signalling could be facilitated through adopting a workflow like mechanism.
- 3) At this point, the virtualized infrastructure has been provisioned and configured. Depending on the mechanism adopted e.g. using virtual appliances for VM creation will have the software / service installed as well. This step focuses on the software level configuration activities. However, if the previous step provisioned the virtualized infrastructure only and no software has been installed, an extra step is required prior to this step which will install the SAP software elements on all the virtual

machines in the landscape. During this software configuration process, SAP stack level parameters will be configured based on the recommendation from the sizing / capacity planning phase. The details of configuration are described in the coming section.

- 4) The SAP landscape is booted up in this step. This requires bringing up the database, application servers, messaging servers etc in the required order. At the end of this step a fully functional landscape should be available for use by the customer.
- 5) After the SAP landscape has been provisioned, configured and booted up, it can be brought subject to a capacity validation benchmarking to ensure that the capacity planning / sizing calculation are accurate and the landscape will be able to ensure the QoS guarantees as agreed in the established SLA.
- 6) If the benchmarking proves that SLA can be supported, the service is ready to be handed over to the customer for use. Otherwise, the cycle must be repeated.

UI Screens:

There are 2 UI screens that are required. These are:

- a) SAP Landscape Configuration (SAP software configuration) Validation
- b) Virtualized Infrastructure Configuration parameters

It would be beneficial if we adopt a workflow like mechanism for disseminating this information to either respective personnel or the respective components. The NW CE can be useful here, we can design this as a process and the respective UI screen will be automatically generated and respective actors will be able to see the required information accordingly with minimum effort.

4.2.2 *Infrastructure Provisioning & Configuration*

This section describes the infrastructure provisioning and configuration procedure in details.

Infrastructure Provisioning

Infrastructure provisioning refers to the activities for creating the virtualized infrastructure which is going to be used for the execution of the software and services. The types of infrastructure we are considering here include virtual machines, networks and storage resources. There are a variety of mechanisms which can be adopted for infrastructure provisioning. Infrastructure providers will choose any of the mechanism depending on the policies or operational procedures. Some of the main mechanisms are listed below which we can consider in our use case for infrastructure provisioning.

- a) Appliance based provisioning: The simplest of all the mechanisms is to use virtual appliances for infrastructure provisioning. Virtual appliances are pre-packaged set of virtual machines with the complete execution environment and application software. Virtual appliances can be described using the standards based methods like DMTF's Open Virtual Machine Format (OVF). OVF based appliances can be readily deployed and configured. This provisioning mechanism can be adopted in our use case easily; however, the hardware resource specification will be dynamically generated based on the service level objectives. This mechanism will most likely be used in our use case demonstration.

- b) VM Pools: This provisioning mechanism requires a pool of standby virtual machines (or number of virtual machines i.e. landscapes) available to be readily allocated to customers. The necessary customer specific configuration needs to be performed however. Adopting this approach would be restrictive though because of the overhead required in maintaining the standby landscapes. However, if the customer requests or the options to choose from are kept to a limited numbers, this option can work well and provide instant provisioning capabilities.
- c) OS Appliance + Software volume pools: This approach works by having the appliances without the application software. Application software is available in the form of storage volumes which can be dynamically attached to the appliances. The advantage of this approach is that storage volumes can be reused of the code and data volumes are maintained separately. Moreover, smart storage allocation mechanism like thin provisioning or Copy on Write can be used for efficient storage consumption.
- d) OS Appliance + SAP Software Installation: In this mechanism, the software is installed from scratch on top of bare bone virtual appliances. This mechanism wouldn't be suitable for most of the situations because of the prolonged time requirement for software installation. However, it could be a solution for custom setup. Most likely, we wouldn't adopt this mechanism for our demonstration.
- e) Custom Setup: This mechanism represents a very specialized setup where everything is installed during the provisioning phase. This mechanism would suffer from the same drawbacks of longer time requirements. However, again this would suite for very specific and customer requirements by the customer. This mechanism is out of scope for our demonstration.

Infrastructure Configuration

The second part of the infrastructure provisioning phase is the configuration of the infrastructure. Depending on the provisioning mechanism used, most of the configuration might be performed during the infrastructure instantiation process. However, the likely configuration that would need to be performed would be at the operating system level configuration. This would include but is not limited to hostnames, IP addresses (in case of static assignments for whatever reasons), HOSTS files, TCP/IP stack configuration (socket buffer sizes, number of open sockets, frame sizes etc.), memory configuration (huge pages), number of processes, number of open files etc.

4.2.3 SAP Landscape Configuration

After the infrastructure has been provisioned and configured, the SAP software related configuration should be performed. Depending on the infrastructure provisioning mechanism, SAP software installation must be performed. However, we assume that the software is installed and ready to be configured according to the performance requirements mandated by the service level objectives. The landscape configuration can be divided into two parts: Application Server Configuration and Database Configuration.

Application Server Configuration

Application Server configuration parameters are defined in the profile files which contain the parameter name / value lists. The profile parameters which impact the performance of the application server can be divided into memory related

parameters and processes/processor related parameters. These configuration parameters need to set before starting up the application servers. The SAP application server doesn't allow dynamic update of these parameter, hence, any change in these parameters require a restart of the application server. These configuration parameters are described as follows:

a) Memory Related Parameters

The main memory related parameters relevant for the application server performance are given in the table below:

Profile Parameter	Unit	Comment
Program Buffer		
abap/buffersize	KB	Size of the ABAP program buffer
abap/pxa		Program buffer sharing mode
Export / Import Buffer		
rsdb/obj/buffersize	KB	Size of export/import buffer
rsdb/obj/max_objects		Max. number of objects in the buffer
rsdb/obj/large_object_size	Byte	Estimation for the size of the largest object
rsdb/obj/mutex_n		Number of mutexes in Export/Import buffer
Roll, Extended and Heap Memory		
ztta/roll_area	Byte	Roll area per work process (total)
ztta/roll_first	Byte	First amount of roll area used in a dialog WP
ztta/short_area	Byte	Short area per work process
rdisp/ROLL_SHM		Part of roll file in shared memory
rdisp/PAGE_SHM	8 KB	Part of paging file in shared memory
rdisp/PAGE_LOCAL	8 KB	Paging buffer per work process
em/initial_size_MB	MB	Initial size of extended memory
em/blocksize_KB	KB	Size of one extended memory block
ztta/roll_extension	Byte	Max. extended memory per session (external mode)
abap/heap_area_dia	Byte	Max. heap memory for dialog work processes
abap/heap_area_nondia	Byte	Max. heap memory for non-dialog work processes
abap/heap_area_total	Byte	Max. usable heap memory
abap/heaplimit	Byte	Work process restart limit of heap memory
abap/use_paging	Byte	Paging for flat tables used (1) or not (0)

Apart from these parameters, some parameters need to be configured at both OS and Application Server level. For example using large pages require configuration at the OS level and the application server profile. At the OS level, we need to do configure the `/proc/sys/vm/nr_hugepages` with the size of the large page which is typically 2048 (2MB). At the application server level the profile parameter `/em/largepages = TRUE` shall be used.

b) Process Related Parameters

These configuration parameters refer to the processes of the application servers. The parameters given in the table below have an impact on the performance of the application server. In addition to these configuration parameters, processor affinity configuration of these processes can also be tuned for better performance.

Profile Parameter	Comment
<code>rdisp/wp_no_btc</code>	Number of Batch Work Processes
<code>rdisp/wp_no_dia</code>	Number of Dialog Work Processes
<code>rdisp/wp_no_eng</code>	Number of Enqueue Work Processes
<code>rdisp/wp_no_spo</code>	Number of Spool Work Processes
<code>rdisp/wp_no_vb</code>	Number of Update Work Processes
<code>rdisp/wp_no_vb2</code>	Number of Update 2 Work Processes

Database Configuration

Database configuration is dependent on the database being used with the application server. Vendor specific configuration parameters will be available for each database. However, for this use case we are going to limit ourselves to the MaxDB database.

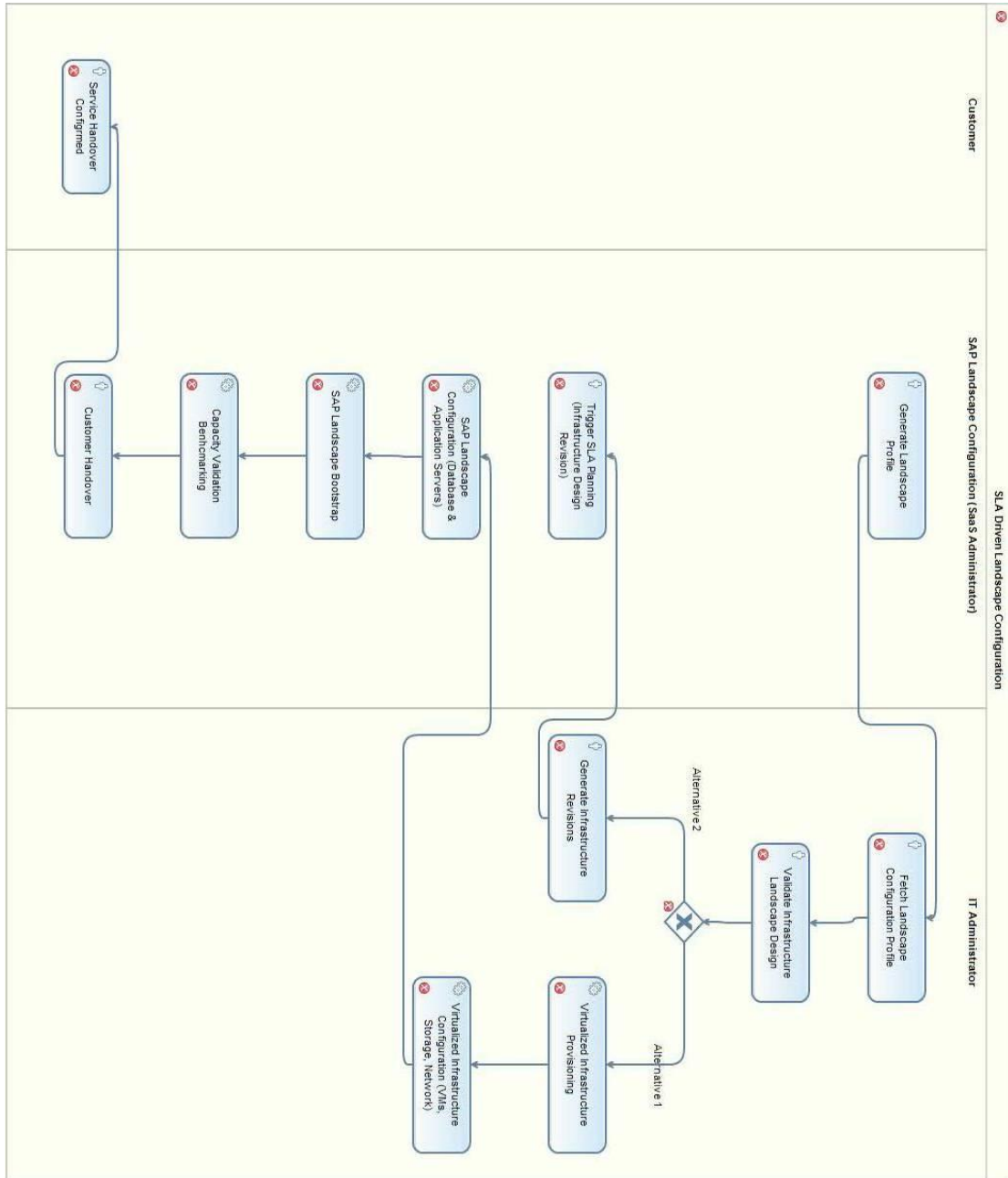


Figure 8: Diagram for SLA Driven Landscape Configuration.

4.2.4 Capacity Validation Benchmarking

At the end of the service provisioning phase, this step can be performed to ensure that the service is able to sustain the OoS guarantees as agreed in the SLAs. Some form of benchmarking can be performed for this purpose. For example if the SLA states that the service must be able to handle 1000 users and the average response time shall not exceed 1.5 sec, a custom SD benchmark can be performed with these parameters and the results can be evaluated to check the performance of the landscape. However, we need to identify the steps required if the benchmark shows that the capacity isn't enough to deliver the service guarantees.

4.3 SLA Driven Landscape Management

This scenario demonstrates the operation and management of the landscape at run time. Together with previous scenarios, an end-to-end process of service planning, provisioning, and management can be demonstrated. The primary focus would be SLAs and the objective is to provision services according to established SLAs. During the lifetime of the service/SLA, the SLA manager undertakes the monitoring and management activities to ensure that the service is performing according to the agreed SLAs. This scenario leverages the previous scenarios and consists of two main building blocks, namely, monitoring and change management.

Given a multi-layered SOA system with multiple abstraction layers and levels, naturally corresponding SLAs are associated with these layers. For instance, process layer, service layer, software/middleware layer, and infrastructure layer. SLA monitoring component needs to show relevant real-time information for different layers. When a potential SLA violation is predicted or a SLA violation event occurs, proper change management procedures need to be taken. The steps of the SLA-driven landscape management process are shown as follows.

4.3.1 Monitoring and Analysis

To meet the expectations of SAP application users, it is necessary to create a monitoring and optimization concept. As SAP has established itself as a solution provider and is no longer just a vendor of Enterprise Resource Planning (ERP) software, the scope of monitoring has changed. Instead of the classic system monitoring, we now talk about *Solution Monitoring and service-level monitoring*, which instead of monitoring just individual system components monitors the business process as a whole across multiple components. Classical system monitoring monitors each software component individually. Due to the number of components that can be involved in a solution, it may be that although each component works correctly on its own, the business process is still not available to the end user with all the efficiency, correctness and security that is required. The cause may be, for example, faulty communication between components. Therefore, solution monitoring must involve business process-specific monitoring, that is, monitoring of component-independent parts of the business process.

Solution Monitoring

Performance indicators and alert information are relevant on different time scales. Generally, we differentiate between two time scales for monitoring:

- Monitoring in terms of minutes and hours (Alert Monitoring)
- Monitoring in terms of days and weeks (Service Level Reporting).

The following components are required:

- System Monitoring
- Business Process Monitoring
- Cross-component Service Level Reporting.

These are described as follows:

System Monitoring

Alert Monitoring covers the monitoring of performance indicators and the automatic recognition of situations that require immediate action.

- Alert Monitoring involves employees from the customer support organization, who have a fundamental understanding of the technology and the application of the e-business solution for which they are responsible. However, they do not necessarily have to be experts. (This results from the fact that alert monitoring must run on a 24x7 basis and that experts from all areas involved are not likely to be easily available.)
- Alert Monitoring is triggered either by a manual error message, such as a telephone call from a user claiming “my application does not work”, or – better – by an alert triggered by the system itself.

Business Process Monitoring

Business Process Monitoring concentrates on alerts which are related to the technical operation of the business processes, such as poor performance of business transactions, processing, hanging or poor-performing interfaces or background jobs. This information is important for day-to-day monitoring as well as for planning the technical optimization of business processes.

Service Level Reporting

The Service Level Reporting reports on the performance of all systems (that belong to a solution landscape), business processes and the support process regularly, for example, on a weekly basis.

By setting clearly defined and verifiable goals, Service Level Reporting enables clear communication between the operator(s) of an e-business solution (these may range from server to database to network, etc., operators) and the customers of the solution that is the owners of the business processes. Service Level Reporting reports on how goals are being met within a defined period of time.

Requirements:

- Service Level Reporting is generated once a week and is oriented to various management areas (IT and business department management). For this reason, it is simple and does not need expert knowledge to be set up.
- Service Level Reporting evaluates performance indicators on management level. These action plans can result from recommendations in SAP Support Services. For this reason, there must be a data connection between Service Level Reporting and these Support Services.
- The action plans are partially pre-configured, yet they can be adapted and expanded to meet the customer’s solution requirements.
- Service Level Reporting allows the customer to generate forecasts and is connected to the predictive Support Services from SAP.

4.3.2 *Change Management*

Change management refers to the procedures and actions taken by the SLA management framework to inject changes into the service or IT infrastructure landscape configuration. The precise nature of actions that should be performed or procedures to be followed depend on the requirements mandated by the change to be implemented. Broadly speaking, change management can be triggered through the following two cases:

1. Due to revised business requirements, the customer revises the requirements and hence a new service level agreement is established through modifications to the already agreed SLA. This course of change implementation follows the

same path as the initial service deployment and configuration except the fact that in this case change requires revising the landscape configuration resulting in some adaptation of resource allocation configuration of the service landscape. Additionally, some service / software level configuration adaptation could also be invoked. The precise nature and scope of adaptation is determined by the SLA planning process as in the initial deployment process. Some of the possible adaptation actions are described later.

2. In the second case, a change is required to sustain the QoS guarantees of services being consumed by the customer. This could be a reactive act triggered through service monitoring and analysis or is invoked as a result of runtime prediction to carry out proactive adaptation procedures. For the moment, we are going to focus only on the reactive change management. During the service monitoring and SLA violation detection process it might be observed that the some SLA are being violated and some adaptation should be applied to the service / infrastructure landscape configuration. To determine the precise scope of adaptation and the actions required to incorporate the adaptation, SLA planning process shall be invoked. SLA planning process should analyze the required adaptation actions to bring the service landscape configuration in compliance to the SLAs in place. It is worth mentioning here that there could be two possible outcomes of the analysis performed by the SLA planning process:
 - a) SLA violation is a result of service landscape configuration problems. In this case, the responsibility lies on the provider side (either service provider or infrastructure provider) and adaptation actions shall be performed on service landscape parameters or infrastructure resource allocation to rectify the situation.
 - b) The second case is that the SLA violation happened as a result of abusive behavior on customer's part in violation to the agreed commitments. In such scenario, the only action that can be performed is that the customer is notified and asked to bring the service usage within the agreed limits.

Apart from the SLA planning and adaptation actions, some other activities can also be incorporated into the change management process. These activities are discussed in the following.

Change Impact Analysis

Changes in a running landscape are most likely candidate to result in service disruption if changes are not very well thought and the impact of changes well understood. An undesired change impact can lead to potential SLA violations resulting in service disruptions and customer dissatisfaction.

It is, therefore, imperative to comprehend the impacts of change to be applied on the running landscape. A smooth and predictable change injection can guarantee sustainable service level guarantees and help enterprises / service providers eliminate any surprising incidents. In order to comprehend the implications of change injection, complete knowledge of the landscape elements is required. This knowledge shall be captured in the form of service knowledge base (as described in the architecture section). The service knowledge base captures detailed information about the landscape elements, the relationships among the landscape elements as well as the dependencies of the individual landscape elements. Based on this, service knowledge base, tools can be developed which can be utilized for simulating change injection into the service landscape. Using these, simulation tools, change impact can be analyzed and a thorough implications of the change

on the landscape can be studied to make sure a smooth and predictable landscape change management process.

Additionally, these change analyses techniques can be used by the administrators to evaluate “what-if” type scenarios for understanding the landscape operations and potential incidents to undertake proactive measure for landscape management.

Change Recording

In order to ensure traceability and audit ability of the operational landscape for SLA auditing, it would be critical to keep track of all the changes that have been applied to the customer landscapes. Service knowledge base can be used to record the metadata about the changes that is about to be applied to the landscape. These change records can be help the service providers in a variety of ways e.g. finding the root cause of a problem, service level auditing for settling penalty claims from the customers etc. It is of vital importance to integrate this change recording practice into the change management process.

4.3.3 Adaptation Actions

As mentioned in the previous section, the change management process might lead some possible adaptation operation on the service landscape. There can be a variety of adaptation possibilities depending on the current status of the landscape as well the required effect on the landscape. However, initially, we will focus on a limited number of adaptations that can be adopted by the change management process. These adaptation actions are described briefly in the following paragraphs.

Resource adjustment

First possible adaptation which can be performed by the change management process is adjustment of resources allocated to the service landscape elements i.e. virtual machines running various pieces of the service landscape. Resource adjustment can further be subdivided into two categories:

- a) Allocating extra resources to the landscape elements. For example adding more processors, memory or disk capacity to the virtual machines. This type of adjustment isn't quite straight forward to apply because some form of cooperation with the underlying operating system is required before these changes can be visible to the software running within the virtual machines. Some operating systems provide hot plug capabilities; however, some manual (automated/scripted) actions still need to be performed. However, in the case of provisioning extra disk capacity to the virtual machine running the database workload can be rather easier to incorporate.
- b) Adjusting the resource share values for the individual virtual machines. Using this adjustment operation, virtual machines can be given priority in the case of resource contention. This form of resource adjustment is easier to incorporate as most of the virtualization management solutions expose interfaces to adjust these values. However, it is worth mentioning that, this type of adjustment wouldn't result in substantial performance improvements, and hence will be used to address minor issues.

Application server reconfiguration

This type of adaptation is equivalent to the one mentioned previously except that it is applicable at the software layer. Application server level resources can be adjusted to cope with the performance related issues. The potential parameters that can be tweaked are described in the landscape configuration section.

Additional application server instantiation

This adaptation action focuses on instantiating new application server instances to accommodate extra workload on the landscape. This is coarser grained action which involved the previously described action implicitly i.e. new virtual machines need to be provisioned with extra resources and the application server instance needs to be configured according to the requirements. The precise nature of resource configuration and application server instance parameters will be determined by the SLA planning process which will identify the requirements to sustain the SLA. This adaptation action will be part of the demonstrator.

4.3.4 Scenario Diagrams

UI requires at least 1 screen for monitoring, with different views showing SLA status for different layers. There is also one screen as management console where proper change management procedures can be triggered.

SAP SaaS administrator (Sam) and SAP IT administrator (Joe) are the main actors in this scenario. On one hand, from an operational perspective, Sam is responsible for the business process level SLA monitoring, which relates to the customer SLAs. On the other hand, Joe is responsible for the technical level SLAs/sub-SLAs, such as monitoring of service, software and infrastructure.

It is worth mentioning here that the actual course of actions and respective entities involved may vary in different situations. The process of change management can be completely automated in an ideal situation and all the monitoring and adaptation actions will be undertaken by software entities configured as per SLA. In other cases, it can be completely human driven where all the activities are performed by human operators, however, it would not be practical for large environments with hundreds of thousands of installations and also taking into account the human error factor as well as the lack of agile response when the process is completely human driven. However, there can some hybrid solutions are more likely where part of the work is done by software entities and human operators are active partially in the change management process.

In this scenario, we are simplifying the storyline and it is more human oriented, however, it can very well be augmented with software entities to increase the level of automation in the change management process. In every step, we add a sub-point to mention other possibilities this can be implemented.

Steps:

- 1) A monitoring console displays all monitored systems/layers and their SLA compliance status. Sam is responsible for the business level monitoring while Joe is responsible for the technical level monitoring.
 - a) As mentioned previously, this can be automated as well by adding software entities for monitoring and issuing alerts to either human operators or other software entities for triggering some adaptation actions.

- 2) Sam detects a business level SLA violation (Average Response Time > 2 seconds for the last 12 hours).
 - a) A Software entity can detect this situation and informs Sam (or some other component) about a possible SLA violation
- 3) Sam checks whether it is caused by the breaking of constraints, e.g. workload excess (number of concurrent users > threshold, use request rate > threshold) at the customer side. If so a notification is sent to the customer and no further action is taken on the provider side.
 - a) In the automated case, the software entity will analyze the situation. Rule can be derived from SLAs and these rules can be validated to understand where the problem lies.
- 4) If the SLA violation is not caused by the customer, Sam starts a hierarchical diagnosis process on the provider side. Firstly he checks if any problem exists at the business process level. If problem is not solved Sam communicates diagnosis requests with Joe on the problem.
 - a) In the automated operations case, if the rule validation yields that the problem doesn't exist on the software side, alert is raised for investigation at the lower levels of the stack i.e. infrastructure.
- 5) A hierarchical diagnosis process recursively propagates the requests to lower layers based on SLA translation rules and dependency graphs. SLA translation module correlates higher-level objectives (e.g. response time) with lower level metrics and parameters (e.g. server utilization, memory usage).
- 6) A cause is identified at a certain intermediate layer by Joe. For example, the reason for the response time violation is that one of the virtual machine running the application servers is overloaded and its utilization is beyond threshold.
- 7) As a measure to cope with this situation, a new instance of the application server needs to be spawned so that the load can be balanced among various instances.
- 8) The resource requirements and configuration details for the new application server instance and underlying virtual machine must be determined through SLA / service planning process.
- 9) Virtual machine with the required resources is provisioned and configured.
- 10) Application server is deployed and configured according to the recommendations from the SLA / service planning process.
- 11) New application server will accommodate some of the workload and the customer SLA should be in compliance. If not an alert is raised to higher management for further instructions.
- 12) If no solution is found down to the lowest level, i.e. the resource layer, it raises alert to higher management for further instructions. Proper re-planning or re-negotiation process will be evaluated.

Besides layered SLA monitoring capabilities for end-to-end processes, there are several key building blocks to realize the above-described scenario. First, SLA translation rules and dependencies have to be developed and specified. For instance, how one SLO level metrics such as response time correlates with one or many lower level metrics? Second, change management procedures have to be well specified, based on instructions or experience.

5 *Conclusions and Future Work*

This document gives a detailed specification of the ERP hosting use case by SAP. In section 2 we introduce SAP's products, technology stacks including NetWeaver, IT foundations, and the standard application benchmark (SAPS), analyse the quality and SLA management related aspects of these and provided a concluding problem statement. In section 3 we give a high-level description of the architecture of the SLA-aware service manager and present the demonstrator system landscape. Built on top of such context, we present three main scenarios of the use case in section 4. Firstly, we introduce SLAs in the SAP landscape sizing and planning. ERP solutions are offered as hosted solutions and we describe how the service provider negotiates SLAs with customers and plan/optimize its landscapes according to SLAs. Secondly, the optimized configuration parameters are used for actual landscape configuration and provisioning. Thirdly, we introduce SLAs in the run-time operation management of landscapes. Two sub-scenarios are elaborated, namely monitoring and change management, and how they relate to each other with respect to SLAs. These three scenarios together form a life cycle to demonstrate holistic SLA planning and management capabilities for hosted ERP solutions.

We plan to further research, prototype, and realize the demonstrator in the coming project years, based on the use case specification described in this deliverable. Firstly, we will investigate the SLA-driven planning and optimization given SLA constraints. At design-time/negotiation time, we want to find out the (sub) optimal sizing and capacity planning parameters for provisioning a system that matches the customer profile at the same time satisfies SLA requirements. Secondly, we plan to investigate SLA-aware monitoring, translation, and change management in a multi-level, service-oriented system. The correlations and mappings of metrics at run time are important topics for further research. Along with the research activities, we will set up the system landscape and start designing and prototyping the service manager. All the research results will be implemented as features/tools in the service manager, and be demonstrated in one of the main demonstration scenarios.

6 *References*

- [CE09] SAP NetWeaver Composition Environment 7.1.
<https://www.sdn.sap.com/irj/sdn/nw-ce>
- [LCE10] H. Li, G. Casale, T. Ellahi. SLA-Driven Planning and Optimization of Enterprise Applications. In proc. of 1st joint ACM WOSP/SIPEW Conf. on Performance Engineering, San Jose, CA, USA, Jan 2010.
- [LTH09] H. Li, W. Theilmann, J. Happe. SLA Translation in Multi-Layered Service Oriented Architectures: Status and Challenges. Manuscript submitted, 2009.
- [NW09] SAP NetWeaver technology platform: Foundation to enabling and managing change. <http://www.sap.com/platform/netweaver/>
- [SAPS09] SAP Standard Application Benchmarks: measuring in SAPS. <http://www.sap.com/solutions/benchmark/index.epx>
- [TK08] W. Theilmann, R. Kilian-Kehr. Quality considerations in SAP Architectures. Proc. of Compuarch08, Springer, 2008.

Appendix A: Glossary

The following list shows the most important entries of the SLA@SOI glossary. Note that terms that are specific for the current document and not part of the overall project wide glossary are marked with an asterix *.

Abstract Service	Any service not yet instantiated (→ <i>service instance</i>) is called an abstract service.
Agreement Initiator	An agreement initiator is a party to a → <i>service level agreement</i> . The initiator creates and manages an agreement on the availability of a service on behalf of either the service customer or service provider, depending on the domain-specific signalling requirements.
Agreement Offer	An offer is the description of the agreement relationship that is sent from → <i>agreement initiator</i> to → <i>agreement responder</i> during agreement creation, indicating the relationship which the initiator would like to form.
Agreement Responder	The agreement responder is a party to a → <i>service level agreement</i> . The responder implements and exposes an agreement on behalf of either the service provider or service customer, depending on the domain-specific signalling requirements.
Agreement Template	An agreement template is an XML document used by the → <i>agreement responder</i> to advertise the types of offers it is willing to accept.
Agreement Term	Agreement terms define the content of a → <i>service level agreement</i> .
Business Service	A business service is exposed/invoke-able via at least some non IT elements.
External Service	External services are exposed across the boundaries of an organization, i.e. across at least two administrative domains.
Guarantee Term	Guarantee terms define the assurance on service quality associated with the service described by the service definition terms. They refer to the service description that is the subject of the agreement and define service level objectives, qualifying conditions and business value expressing the importance of the service level objectives.
Infrastructure Provider	A specific kind of service provider who focuses on the provisioning of → <i>infrastructure services</i> .
Infrastructure Service	An infrastructure service is a specific → <i>IT service</i> which exposes resource/hardware-centric capabilities.

Internal Service	Internal services are exposed within the boundaries of an organization, i.e. within one administrative domain.
IT Service	An IT service is exposed/invoke-able by means of information technology. Specific classes of IT services may be software services, infrastructure services or media services.
Offered Service	An abstract service which is offered by a specific → Service Provider to its → <i>Service Customers</i> .
Service	A means of delivering value to Customers by facilitating Outcomes Customers want to achieve without the ownership of specific Costs and Risks. See also → <i>service interface type</i> , → <i>service stage</i> , → <i>service exposure</i>
Service Stage	The stage a service reaches over time from fully abstract to actually instantiated. See also → <i>abstract service</i> , → <i>offered service</i> , → <i>service instance</i>
Service Consumer	Person(s) who actually consume/use the provided services. Typically they belong to the → <i>service customer</i> .
Service Customer	Someone (person or group) who orders/buys services and defines and agrees the service level targets.
Service Description Term	Service Description Terms describe the functionality that will be delivered under the → <i>service level agreement</i> . The agreement description may include also other non-functional items referring to the service description terms.
Service Exposure	Services can be exposed either internally (within the same administrative domain) or externally. See also → <i>internal service</i> , → <i>external service</i>
Service Instance	A concrete implementation of an → <i>offered service</i> which is ready for consumption by service users.
Service Interface Type	Describes the nature of an actually exposed service, i.e. about the nature of his invocation interface. See also → <i>business service</i> , → <i>IT service</i> , → <i>composed service</i>
Service Level Consequence	An action that takes place in the event that a → service level objective is not met.
Service Level Agreement	An agreement defines a dynamically-established and dynamically managed relationship between parties. The object of this relationship is the delivery of a service by one of the parties within the context of the agreement. The management of this delivery is achieved by agreeing on the respective roles, rights and obligations of the parties. The agreement may specify not only functional properties for identification or creation of the service, but also non-functional properties of the service such as performance or availability. Entities can dynamically establish and manage agreements via Web service interfaces.

Service Level Objective	Service Level Objective represents the quality of service aspect of the → <i>agreement</i> . Syntactically, it is an assertion over the → <i>agreement terms</i> of the agreement as well as such qualities as date and time.
Service Provider	An organization supplying services to one or more internal customers or external customers.
Software Provider	An organization producing → <i>software components</i> which might be used by a → <i>service provider</i> to assemble actual → <i>services</i> .
Software Service	A software service is a specific → <i>IT service</i> which is exposed/invoke-able by means of software entities such as Web services, user interfaces, or software-based business processes.
Software Component	Software components are the entities produced at design-time by a → <i>software provider</i> .

Appendix B: Abbreviations

CMDB	Configuration Management Database
ERP	Enterprise Resource Planning
NFP	Non-functional property
NW	NetWeaver
SaaS	Software as a Service
SLA	Service Level Agreement
SOA	Service Oriented Architecture
TCO	Total Cost of Ownership