PaaSage

Model Based Cloud Platform Upperware

Deliverable D6.1.1

Initial Requirements

Version: 10

PROJECT DELIVERABLE

Name, title and organisation of the scientific representative of the project's coordinator:

Mr Pierre Guisset, Tel: +32 -4-96869019, Fax: +33-4-92385011
E-mail: Pierre.guisset@ercim.eu

Project website address: http:/paasage.eu/

Project	
Grant Agreement number	317715
Project acronym:	PaaSage
Project title:	Model Based Cloud Platform Upperware
Funding Scheme:	Integrated Project
Date of latest version of Annex I against which the assessment will be made:	29th August 2012
Document	
Period covered:	M1-6
Deliverable number:	D6.1.1
Deliverable title	WP6 Initial Requirements
Contractual Date of Delivery:	31/3/2013
Actual Date of Delivery:	10/4/2013
Editor (s):	Christophe Ponsard (CETIC), Philippe Massonet (CETIC), Philipp Wieder (GWDG), Dirk Muthig (LSY), Stefan Spahr (LSY)
Author (s):	Erich Schelkle (ASCS), Jenny Kremser (ASCS), Franky Vanraes (BEWAN), Jonathan Demortier (BEWAN), Christophe Ponsard (CETIC), Philippe Massonet (CETIC), Espen Sjøvoll (EVRY), Philipp Wieder (GWDG), Dirk Muthig (LSY), Stefan Spahr (LSY), Tao Jiang (USTUTT-HLRS), Anthony Sulistio (USTUTT-HLRS)
Reviewer (s):	Pierre Guisset (ERCIM) and Brian Matthews (STFC)
Participant(s):	ASCS, BEWAN, CETIC, ERCIM, EVRY, FLEX, GWDG, LSY, SINTEF, STFC, USTUTT-HLRS
Work package no.:	WP6
Work package title:	Requirements, Integration and Testing
Work package leader:	Stephane Mouton (CETIC)
Distribution:	PU
Version/Revision:	10
Draft/Final:	Final
Total number of pages (including cover):	75

DISCLAIMER

PaaSage is a project funded in part by the European Union.

History

Date	Version	Author	Description
14/2/2013	1	D. Muthig	Initial structure after Berlin Meeting
20/3/2013	2	C. Ponsard	Refined structure with outlook after Frankfurt Meeting
28/3/2013	3	S. Spahr	Compilation of the final document
28/3/2013	4	C. Ponsard P. Massonet	Polishing
29/3/2013	5	P. Guisset	Review
31/3/2013	6	P. Wieder	Revised Section 1 according to Pierre's review.
3/4/2013	7	B. Matthews	Review
03/4/2013	8	S. Spahr	Revised document according to Pierre's, Philipp's and Brian's review.
9/4/2013	9	S. Spahr	Include changes from GWDG, BEWAN, EVRY, HLRS/ASCS and CETIC
10/4/2013	10	S. Spahr	Reformatting, correct typos etc. Relocated section 'Organization behind the case' in front of each chapter. Final version.

Table of contents

1 Introduction

This deliverable describes the objectives, requirements, and scenarios for future usage of Cloud environments. The requirements have been elicited from the three PaaSage use case sectors as there are industrial cloud, eScience, and the public sector. The requirements analysis has been carried out to specify them in such as way that they can be tested and validated by PaaSage use case stakeholders.

The main goal of this document is to capture the requirements that different potential adopters of cloud technology will have about the way to deploy new applications or migrate existing applications onto a cloud. To reach this goal, this deliverable will describe the objectives, requirements, and scenarios for future usage of cloud environments. The presented requirements are elicited from four PaaSage use cases originating from three aforementioned different application domains:

- Flight scheduling (industrial sector)
- Industrial Enterprise Resource Planning (industrial sector)
- Electronic portal for citizen-city (public sector)
- Resource intensive simulations including the automotive domain (eScience sector)

A complementing goal of this deliverable is to capture the requirements in a way that they can be tested and validated by PaaSage use case stakeholders. In order to achieve this, each specific use case is mapped onto the general abstract PaaSage workflow. Through this step, the requirements are, on the hand, explicitly mapped onto the different steps of the generic workflow designed by PaaSage, and it is, on the other hand, possible to provide an early assessment of the validity and general applicability of the approach proposed by PaaSage.

Furthermore, the use cases described here and the requirements gathered are the foundation for the realisation of the demonstrators developed by WP7. Following the aforementioned sector-related structure, the demonstrators will show the applicability of the PaaSage system. Depending on the use case demonstrated, different key feature, like application optimisation or process interaction, are in the focus of a particular demonstrator.

Note this deliverable is not about the detailed requirements specifications of components to be developed within technical PaaSage tasks. Such work will be carried within the PaaSage work packages WP2-5 in compliance with the architectural guidelines defined by WP1.

The deliverable will detail each of the use cases listed above. It is structured as follows:

- The general template structure used for each use case is described in Section 2.
- Sections 3 to 6 detail each use case listed using this template.
- Section 7 gives a synthesis consolidating, structuring, and highlighting common requirements across the cases.
- The final section explains the next steps that will be performed within WP6 to refine the requirements, match them with the WP1 architecture and specific components, and later on validate the produced prototypes against the requirements.

The deliverable at hand documents use case-specific requirements that are the foundation of the technical work executed by WP2 to WP5. As the PaaSage system will evolve, it will be necessary to re-assess some of the requirements and match the use cases with the architectural and technological decisions and developments. This deliverable will therefore reflect this evolution, being used project-internally as a “living-document”. The final requirements deliverable of WP6, D6.1.2, will document the results accordingly.

2 Use Case Structure

We describe here the general template that will be followed by each use case.

2.1 *Organisation behind the Case*

This section will describe the organisation of the company presenting the case. The description relates to the general organisation described in the overview report. How are the roles realized in your organisation? Where are organisational boundaries? What is the competence or responsibility of the actors in real life? Are there other processes in your organisation that overlap or interact with the PaaSage workflow etc.

2.2 *Objectives*

This section aims at describing what each company is doing in general; what are the classes of products or processes which can be improved by using cloud computing in general and especially by using the PaaSage method.

2.3 *Current Status (as-is)*

This section will give a description of the current status of the selected case.

2.4 *Target Picture (to-be)*

This section will describe the improvement which should be reached by using cloud computing together with the PaaSage method.

2.5 *Walkthrough PaaSage Workflow*

In this section a case-specific walk through the general PaaSage workflow will be described. Important driving questions are: Are there any specific requirements and constraints in the context of individual steps? Do you use specific tools or technologies? Which steps are the most important or critical ones? How could the platform make a significant difference compared to today's practices?

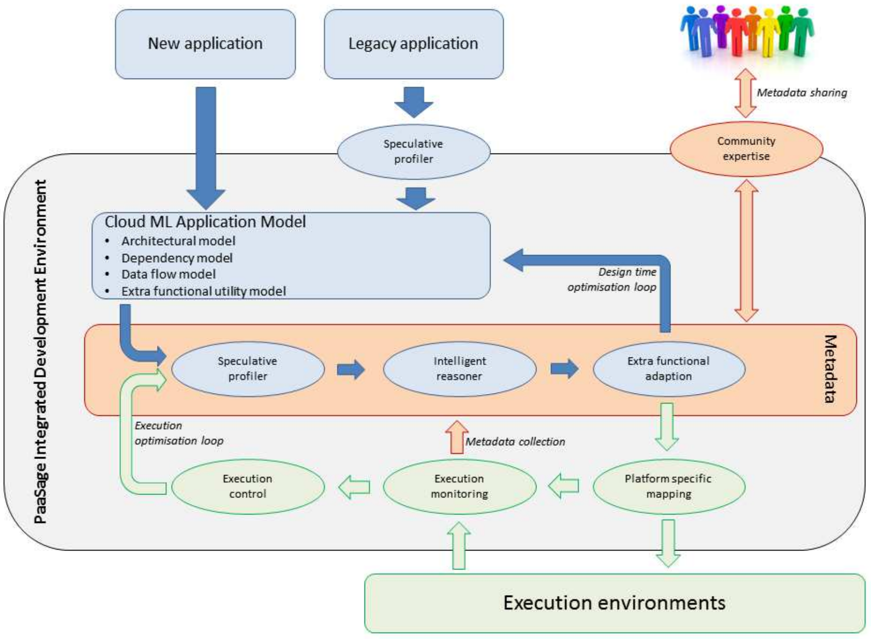


Figure 2-1. PaaSage abstract workflow.

3 Industrial Sector Case – Flight Scheduling

This case is supported by Lufthansa Systems (LSY).

3.1 Organisation behind Case

3.1.1 The Company

Lufthansa Systems provides consulting and IT services for selected industries and has a leading position in the global aviation industry. The wholly-owned subsidiary of the Lufthansa Group offers its customers the entire range of IT-services, including consulting, development and implementation of industry solutions as well as operations.

At its headquarters in Kelsterbach near Frankfurt/Main, Germany, the company operates one of the most modern data-centres in Europe. Lufthansa Systems has offices in Germany and 16 other countries and employs about 3,000 people.

3.1.1.1 Airline Solutions & Services

Lufthansa Systems offers airlines of all sizes a variety of solutions for controlling and optimizing their core processes. Network carriers, regional airlines and low-cost carriers will all find packages of solutions tailored to their individual needs.

Lufthansa Systems offers integrated solutions for managing cost-effective and safe flight operations. The portfolio of solutions ranges from network planning, operations control and crew management, to hub management and load control. Our state-of-the-art products and services for ground handling fulfil the demands of today's business.

Today, about 200 airline customers worldwide rely on the Lufthansa Systems portfolio of products and services.

3.2 Objectives

3.2.1 Selection of a use case scenario

From the wide variety of airline applications Lufthansa Systems offers, we selected an application from the NetLine product suite, which is used for airline schedule planning, it's called NetLine/Sched.

Today's airlines need to permanently revise their schedule plans in response to competitor actions, or to follow updated sales and marketing plans, while constantly maintaining operational integrity. This makes schedule management a very complex process. These challenges call for a state-of-the-art scheduling system which optimally supports the development, management and implementation of alternative network strategies. NetLine/Sched supports all aspects of schedule development and schedule management. It offers powerful and easy to use schedule visualization and modification, supports alternative network strategies and schedule scenarios and measures the profitability impacts of alternative scheduling scenarios. The system is used every day by more than 30 airlines around the globe, ranging from small to large carriers and using different business models.

3.2.1.1 Overview of NetLine/Sched - Schedule Management Solution

Figure 3-1 shows the application timeframe for schedule planning. It starts around 6-24 months before the day of operation (medium-range planning), over a short-range and implementation phase (~1 month before day of ops) into the operational phase of the schedule. Even though operation control is supported by a different application (NetLine/Ops) the scheduler is still involved in these activities, because scheduling is a never ending and repetitive task and the end of one schedule is the start of another one …

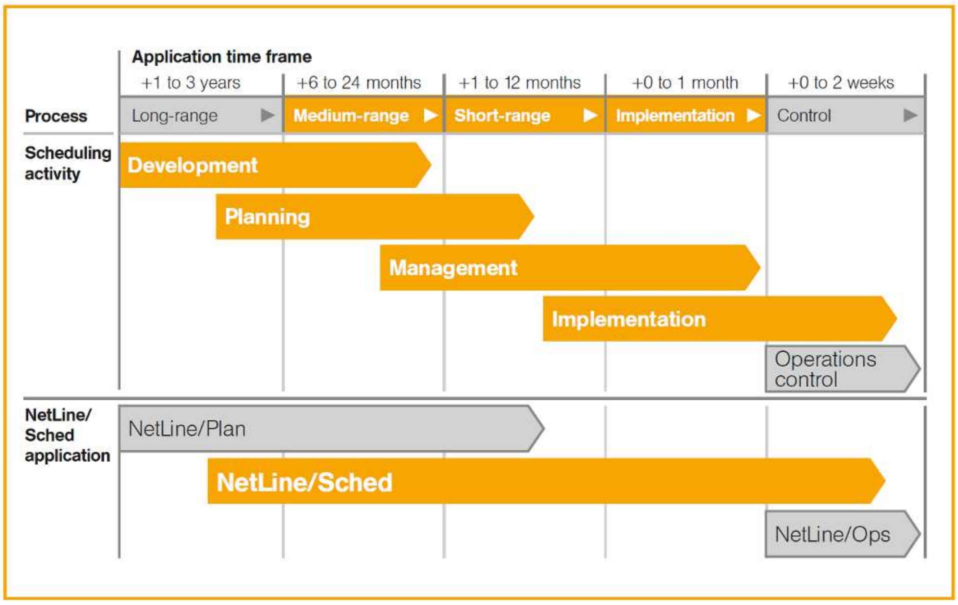


Figure 3-1: A typical scheduling process

Flight schedules consist of single legs (to travel form 'A' to 'B'). These legs are chained into so-called *aircraft rotations*.

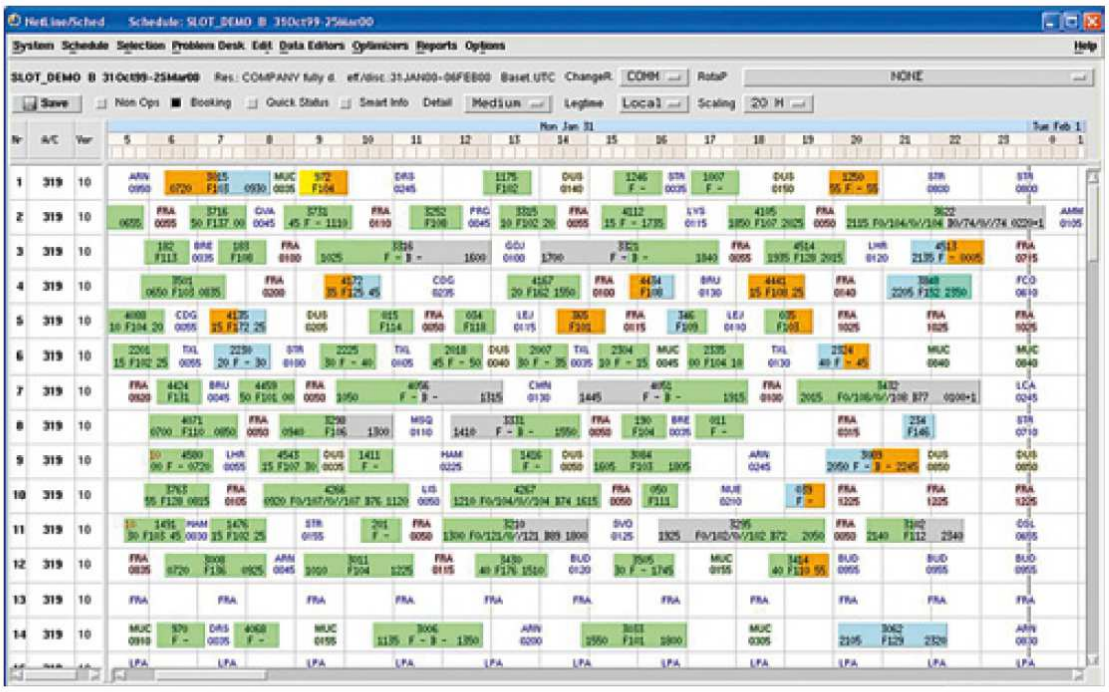


Figure 3-2 : NetLine/Sched Aircraft Rotation view

One of the major key issues of the schedulers work is to realize the planned flight schedule of an airline with a minimum number of physical aircrafts used. Together with a minimum of (unproductive) ground time. Constraints are necessary aircraft check events, aircraft maintenance operations, airport restrictions etc.

Or, to summarise, the objectives of flight scheduling are:

1. Maximize the schedule profitability
   - Maximize the aircraft utilization
   - Ensure a high seat load factor
2. Keep the schedule feasible.

The following example illustrates the number of entities which needs to be handled:

- A flight schedule is normally built for one season, summer or winter schedule.
- The schedule is built-up on aircraft rotation (and an aircraft rotation is not limited by such a schedule season; it laps in and out of the current schedule period and is nearly infinite, i.e. from purchase until decommissioning).
- This seasonal schedule spans over 6 month.

For a mid-size airline such a schedule can contain ~200.000 events (legs, checks etc.) and for a large airline it can be over 1 million events!

Other tasks which are executed by schedulers with support of NetLine/Sched are:

- Check schedule consistency
- Create schedule scenarios
- Schedule administration
- Slot management
- Schedule optimization
- Schedule simulation
- Profitability evaluation
- Reporting.

To execute these tasks efficiently the flight scheduler needs access to other IT-systems within the airline and outside of the airline. Examples are:

- Collect booking figures and forecasts, e.g. from a revenue management system
- Publish schedule information / schedule changes
- Collect airport slot information
- Collect data from preceding and subsequent applications of the schedule planning workflow (see also Figure 3-1).

3.2.2 Motivation for the Cloud

From year to year the airline industry has the challenge of working more and more cost effective. Cooperation's and mergers happen, to use the synergetic effects and to establish the necessary market power.

To meet these challenges, the airline companies' needs, amongst other things, an IT infrastructure, application landscape and system operation with high flexibility and usability. The applications must support different kind of collaboration models, better than today.

To support such strategic alliances of individual airlines (i.e. former competitors) the companies needs the aforementioned flexible infrastructure and application software. These environments must be able to scale vertically and horizontally very well. Therefore, besides the infrastructure, the used application software must be designed to scale and to use the given resources very efficiently.

Cloud computing will be one of the key factors to realize this flexibility. A company which develops application software to run in a cloud environment needs abstraction from specific cloud service providers to prevent a vendor lock-in, to allow shorter development cycles for new products and to gain additional benefit for the application user by providing additional features.

The following chapters highlight the aspects of system operation and application development in more detail.

3.2.2.1 System Operation

As previously mentioned Lufthansa Systems offers its customers the entire range of IT-services, including consulting, development and implementation of industry solutions as well as operations.

Depending on the kind of the IT service provided to e.g. a NetLine/Sched customer the organisation and the roles servicing the customer are slightly different.

Possible operator models provided to a customer:

- The customer has an own IT department and operates the application on-premise in its data centre. Therefore the airline buys an application license and the product is introduced within a jointly implementation project. The customer gets application-only support by the Lufthansa Systems application customer support department.
- Lufthansa Systems operates the whole system for the customer. The customer doesn't need an own data-centre. But he still owns an application license. The implementation project has a smaller scope from the customers' viewpoint, because of the outsourcing of the operation. The customer gets full application and system operation support by Lufthansa Systems.
- Lufthansa Systems operates the whole system for many customers. The customer doesn't need any special IT nor does it buy an application license. There will be a time based service agreement for the application and the implementation project focuses on the user training. Lufthansa Systems gives full application support.

There are advantages and drawbacks for all these operator models. However, cloud computing will strongly influence these operator models and the processes and roles associated with. Therefore it will be also a target picture of the organisation behind this PaaSage use case.

From operation point of view the significantly reduced costs (more as it can be realized by pure virtualization) is a major issue. This reduction will be realized through a homogenous infrastructure by using cloud platform standards. Using the PaaSage method enables us to realize these factors also across different cloud infrastructures. Supporting deployment into hybrid clouds easily (build up on customer and provider cloud infrastructures) is another key benefit of the PaaSage method.

This homogenization of the infrastructure might be the basis for a homogeneous application landscape. This in turn evolves consolidated processes around.

Lowering the heterogeneity of the infrastructure and the application landscape as well as the process diversity has a direct impact on the staff structure. There is less special qualification for people needed and due to automated control of the operation there is even less personal needed at all.

3.2.2.2 Application development

Beside the operation model and the services provided to the customers, the application development is a huge part of the Lufthansa Systems portfolio. The offered software products are flexible and highly customizable. They share data with other products whenever it makes sense.

Developing applications which are designed to run in a cloud environment will benefit from at least these topics:

- Reduced complexity
- Improved quality
- Reduced development time / reduced cost

Reduced complexity

Modularization enables us to develop in a feature-based approach. Subsystems and services are then more decoupled and well documented and therefore the demand to know every part of the system is lower than today.

Operational aspects are hidden by the cloud architecture. Standardized persistence models can be offered by the cloud environment and used by a service. Scalability is inherently supported by the cloud infrastructure if the application service is designed according to the cloud design patterns.

Also currently necessary support of different operating systems (in parallel, e.g. IBM AIX, different flavours of Linux, Solaris …) for customers operating an own data-centre can be reduced by using the virtualization approach behind cloud computing.

Improved quality

Today, to setup and to maintain an (integration) test environment for such a complex application environment as NetLine/Sched can be hard and painful.

Several connections to other systems (realized as a mock-up or as test instances, too) needs to be configured, database content from production instances must be loaded into separate test database instances etc.

Future test scenarios gains from a better modularization as well as from the cloud infrastructure itself.

Modularized systems might be tested in a down-scaled test scenario (before the integration test is executed). Only changed services needs to be tested by the developer and/or the test team.

Provisioning of an adequate test environment should be considerable easier in a cloud infrastructure than configuration of a, non-virtualized, conventional environment.

Reduced development time / reduced cost

The aforementioned change of test execution of modularized systems is also reflected by the development process. A more iterative process model is supported by such service oriented architecture. The feedback loop between requirements analysis, prototyping and the customer is much more agile than before.

This will result in shorter development cycles and therefore the project can be finished with reduced cost.

3.3 Current Status (as-is)

3.3.1 Overview

Today, the system consists of a relational database server (RDBMS) to persist the schedule data and the supplemental basic data (e.g. airport details, aircraft configurations, constraint configurations etc.) and a fat client installed on a (separate) server (see Figure 3-3).

The fat client reads the data from the database and presents the schedule information to the user (using a graphical UI and by different reports) over a virtualized desktop sharing solution (like OSGD).

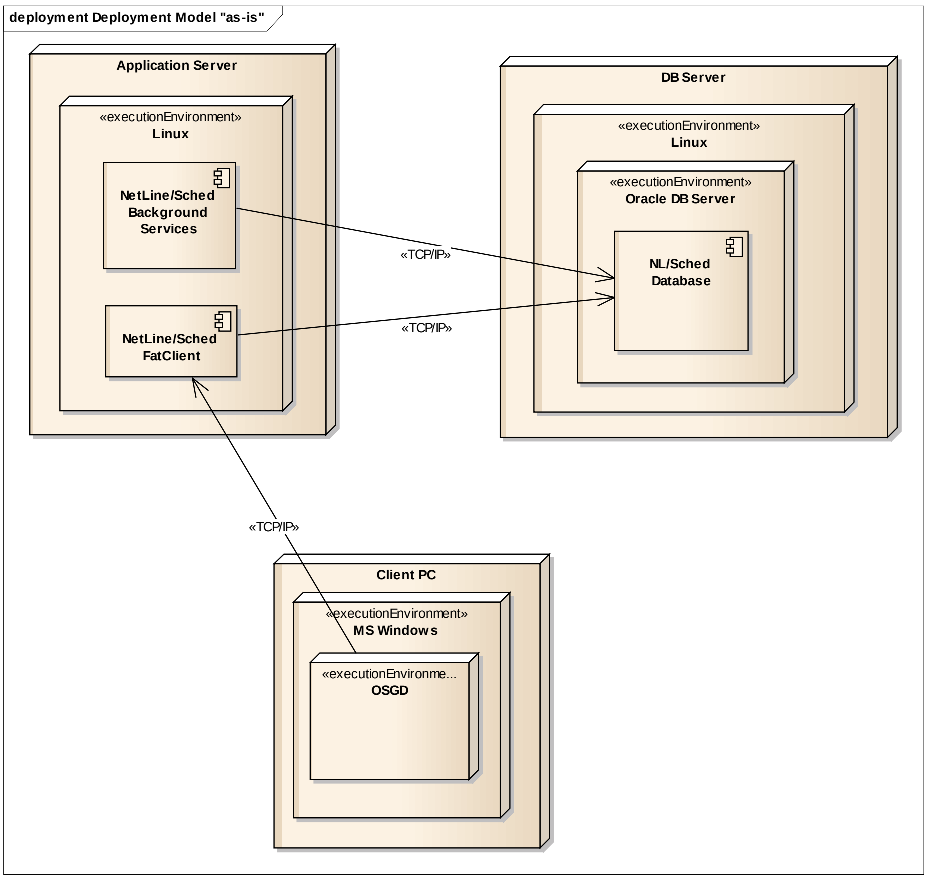

deployment Deployment Model "as-is"
Application Server
«executionEnvironment» Linux
NetLine/Sched Background Services
NetLine/Sched FatClient
DB Server
«executionEnvironment» Linux
«executionEnvironment» Oracle DB Server
NL/Sched Database
«TCP/IP»
«TCP/IP»
«TCP/IP»
Client PC
«executionEnvironment» MS Windows
«executionEnvironme... OSGD

Figure 3-3 : Current deployment model of NetLine/Sched

The user (scheduler) uses the graphical user interface to modify the schedule, to run (complex) checks or to compare his/her schedule with the other versions stored in the database. The reporting engine is used to create detail lists, summaries and business analysis reports.

The RDBMS and the application server(s)[1] for the fat clients, additional service processes etc. are operated in our data centre or in a customer's data centre.

The sizing of the servers must be done up front. Adding new server is costly, needs special setup and installation of software and it is sometimes a very complex task (e.g. transform an Oracle single node database server into a cluster (RAC) database server).

3.3.2 Detailed View

The current solution has limitations. The following subchapters focus on them.

3.3.2.1 System configuration and operation

The system must be ordered to handle the biggest possible workload in an acceptable time. This means a waste of expensive computing power for the remaining period.

Complicated forecasts and estimations are necessary to create a system design which matches as much as possible the customers needs for the next 3-5 years of the system operation. To prepare the configuration for the unknown system load for the future, this is every time a balancing-act between cost effectiveness and over-sizing.

Hardware upgrades, e.g. due to an underestimated sizing in the past or to handle additional partner or sister airlines are expensive and need a (mostly complicate) migration procedure. These migration steps needs to be executed in a test environment beforehand, which results in a deferred implementation of the whole project paired with additional costs.

Even so, the implementation of such an (additional) test system or the update of an existing test system e.g. to the current productive version of the software/database etc. is a costly and time-consuming task.

Some of these imponderables are handled by using virtualized environments in our data centre. But there are still challenges, waiting to be solved.

3.3.2.2 Functional enhancement

The monolithic design of a fat client makes it harder to enhance existing functionality or to add new functionality. For such tasks the system designer and the developer needs a thorough understanding of the whole system and the interdependencies of each component.

3.3.2.3 Flexible user interfaces and usability aspects

The fat client architecture, using memory cached data structures per client instance, might give a good performance in respect of the interaction with the user. But on the other hand the tight integration of data structures, business functionality and the graphical user interface (GUI) makes it nearly impossible to attach modern user interface techniques to the functional core of the system.

Due to unsupported interaction models (like gesture control) it is impossible to re-use the existing GUI components together with mobile clients. Also a browser based GUI solution is far beyond the means.

[1] 'Application Server' is not meant in the sense of a Java application server, like JBoss AS, but as a hardware resource running the application(s), e.g. a Linux server.

Besides the missing integration into latest GUI technologies, the current solution lacks of other usability issues.

For the user it is impossible to share data between client sessions. Because the fat client collects all data in-memory, this data is not accessible from other sessions; not even for the same user id. Therefore the collaboration with other users is complicated and only possible by using a different medium to communicate (verbal communication, copy-and-paste, email, paper).

Due to the facts described above (see Functional enhancement) the availability of a new application feature is only possible after completion of a complex rollout of the complete application. This does not fit with the experience users have from their Smartphone, where it is possible to quickly solve a problem by installing an App.

3.3.2.4 Data Management

The data management is done by using a centralized (relational) database server. It follows the strict consistency model. This hinders also to follow new trends and to implement changes in the working environment of an airline customer.

3.4 *Target Picture (to-be)*

3.4.1 Overview

The fat client solution should be replaced by different, modern and flexible client solutions.

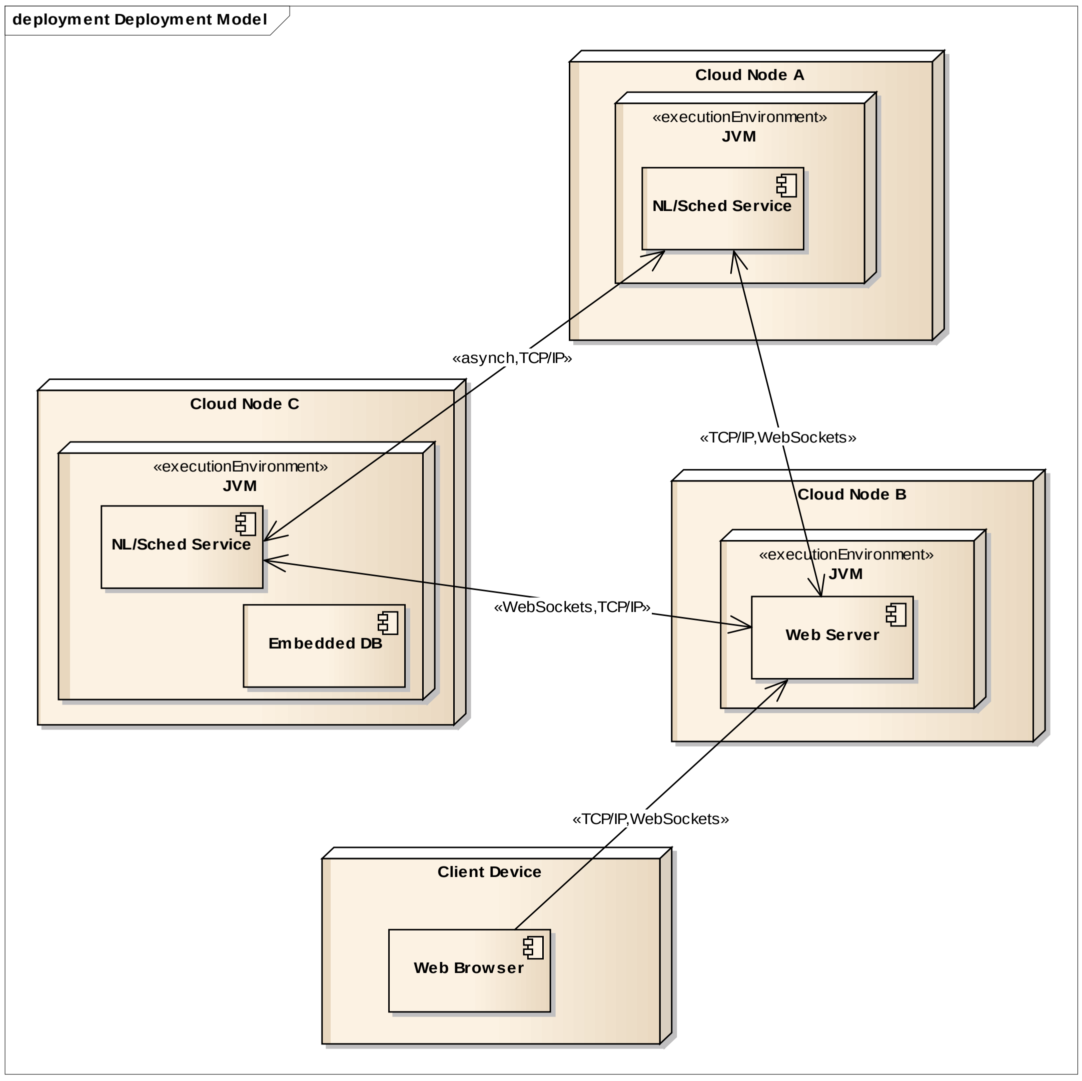


Figure 3-4 : Future deployment model of NetLine/Sched

Not all users of such a scheduling solution need all functionality every time.

Users have different roles (maybe over time), different knowledge about scheduling insights (e.g. expert schedulers vs. supporting staff) and also different environments where they work. A scheduler can e.g. work in his/her office using a full-fledged power client or he/she can be in a meeting and needs just read-only access to the data over a mobile device.

Therefore the to-be solution should be no longer monolithic and database centric. The system must be scalable in the sense of number of users and also in respect of computing power for sessions with more demand (e.g. for automatic optimizations).

This requires cutting down the system into smaller services which can be distributed in a cloud environment and used by different clients.

Security concerns must be covered at all time, e.g. moving from a private cloud into a public cloud (even for parts of the system) must be possible in a secure and reliable way.

Access to external interfaces is a vital part for such an application. Schedules can be exported and imported using a standard file format (SSIM format). And schedulers can trigger sending information to other departments or to partners of the airline.

High availability is important for airlines. For the schedule planning solution this is definitely the case when different user groups (e.g. partner airlines), operating in different time zones around the world are using the (same) system; i.e. also multi-tenancy is a must.

3.4.2 Detailed View

3.4.2.1 System configuration and business operation

In case of system operation, elasticity is requested by default. This demands that needed computing power is available during peak times, even without announcing it before. In return this implies a pay-per-use model, to avoid the waste of money for unneeded computing power during idle periods.

Using cloud services helps the customer to get rid of an in-house hosting service and to concentrate on the core business of an airline – mainly flight operation.

The consequence is that the airline

- needs fewer or even no IT specialists of its own
- reduces the complexity to run the business
- reduces the costs for on premise infrastructure, IT maintenance, software licenses, personnel, computing centre security, complex troubleshooting tasks etc.

Further benefits of operating airline business applications in a cloud environment are:

- Increased interoperability, at least for applications of the same application suite.
- Easy setup of different test environments for different test scenarios (e.g. RfC tests, exploration of new business scenarios, integration tests etc.).
- Test systems are as close as possible to the real application, but still strictly separated; just another instance in the cloud.
- Pay per use, also for such test environments

The key benefits of Scalability through the usage of cloud computing are:

- Flexible business operation. If the company growth or a merger happens, the application operation grows also.
- In conjunction with the term 'Interoperability' it is easier to cooperate with partner or sister airlines.

3.4.2.2 User interfaces and usability

The application users await several improvements from a cloud based solution.

This new model supports a kind of a 'transient workflow', which means that everything the user does is persistent and available on whatever client he/she works on. When a user moves e.g. from the desktop browser to a mobile client, he/she expects to see the same data after login to the same application.

This workflow is heavily supported by using new client technologies like web browser, mobile clients etc.

A basis to enable such a workflow is that the application performance is independent of the client hardware and the separation of the visualization and the core application services. Thus, users can get new application functionality on top of existing services, e.g. by installing small Apps with dedicated, small functional components.

These small Apps with additional features or updates of the application benefits from the cloud deployment model to be quickly available on demand.

As said before, performance is independent of the used client; however it depends on the network connection into the cloud. The new architecture should use cloud specific network optimizations like near edge service relocation to provide the fastest network access, independent of the users current location. This feature supports the collaboration of distributed teams in an excellent way.

And collaboration with colleagues at all is easy, because the internal status and the data can be shared instantly.

3.4.2.3 Data Management

Cloud enabled applications need a different approach to store the application data. Services in the cloud should be work in a stateless way, because a failure of a service or the drop out of a whole node can happen more often than in today's high availability (cluster) environments.

Therefore, also the database technology used in a cloud environment needs to be a different one. Topics like the CAP theorem, ACID vs. BEST, the shared-nothing approach etc. needs to be addressed in such application architecture, designed for the cloud.

3.5 *Walkthrough PaaSage Workflow*

3.5.1 Deployment Preparation

Deployment preparation contains all steps necessary to create an application bundle which can be transferred into and executed by a specific cloud environment.

This step covers the upper part of the PaaSage Workflow (see Figure 2-1).

There are different kinds of execution environments possible. At one hand we have so-called *logical environments*, including:

- development environment
- test environment (also for UAT)
- production environment

And we need of course *physical environments*, including instances of a

- private cloud
- public cloud
- hybrid cloud

These physical environments can also be grouped by functionality, like:

- a database cloud
- a development cloud
- a customer cloud
- an alliance cloud

The logical environment is realized by identifying specific modules and by defining in which physical environment they need to be deployed.

The cloud topology must be definable (e.g. like it is possible with OASIS TOSCA, see https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca)

The following key issues must be fulfilled:

Maintain full portability

- No vendor lock-in
- flexible deployment
    - o during runtime
    - o development / test / production
- platform independent; that could also mean your cloud infrastructure changes (even if it is your own cloud)

Security concerns must be considered

- Requirements from the Chief Information Security Officer (CISO) must be implemented
- customer requirements (e.g. from a worker council)

- legal requirements (country-specific)

Access to external interfaces (specified via CloudML per business services or service cluster or module)

- cloning
- network setup
- flexibility

Availability must be calculated down to services and then realized as requested

- online / offline
- downtime
- 24/7

3.5.2 Execution and Operation

This part of the PaaSage workflow (the lower, green bubbles of Figure 2-1) covers all topics around the execution and operation of the application.

There we need a platform-specific mapping:

- for all physical environments
- which realizes the logical picture in a physical way (it gets alive)
- containing all administrative access rights to resources in the cloud necessary must be realizable in a provider-independent, unified way; e.g.:
    - file system access
    - database access
    - ssh key configuration

Besides the platform specific mapping we need support for execution monitoring and control. Execution monitoring covers:

- Measurement and collection of data from the physical environments and a mapping it to the logical views
- Support of standard monitoring tools and maintenance (to integrate operation smoothly with current operational practices)

Execution monitoring (and operation) must be done by existing teams (24/7).

Execution control analyzes monitoring results and derives requirements etc. to be fed into next deployment preparation cycle. It combines different logical views; combines logical views from different products, combines reports from same physical environments etc.

This is done in a bi-directional way, with a link to the community:

- feed in your experience
- consume experience from others to draw your conclusions or decide on next steps

And also company internal 'communities' might share data with the execution control part of PaaSage, e.g. service management brings in market perspective (like customer plans a sales promotion and needs more capacity; Operation detects an irregular operation which requires fast re-calculation, etc.)

All aspects of Scalability must be fully supported:

- Core service definitions (see also http://www.linked-usdl.org/ns/usdl-core)
    - Elasticity, which includes:
        - Maximum processor cores
        - Maximum main memory
        - Maximum external storage
    - scale-out, which includes:
        - Maximum nodes to be allocated
- Accounting, which includes:
    - Pricing (see also http://www.linked-usdl.org/ns/usdl-price)
- Service levels (see also http://www.linked-usdl.org/ns/usdl-sla), which includes:
    - Guaranteed state and action
    - Service level profile

4 Industrial Sector Case – Industrial ERP

This case is supported by the BEWAN partner.

4.1 Organisation behind the Case

BEWAN is an IT Service company located in Belgium, delivering products and services in the domain of IT infrastructure, software development and consultancy.

BEWAN has two offices in Belgium, in Flanders (Wevelgem) and in Wallonia (Waterloo). Headcount is 63: architects, developers (analysts & programmers), support people, system engineers and off course SG&A people.

BEWAN carries out ICT projects for a wide range of customers, mainly Medium sized companies, but also local departments of multinational companies. BEWAN develops projects in the domain of

- IT infrastructure: BEWAN advises, sells, installs, implements, offers managed services… in the domain of hardware, such as servers, networks, security, hosting, workstations (PC's), peripherals…
- ERP software: (enterprise resource planning): BEWAN advises, sells, customizes, delivers, implements, gives training & support… in the domain of ERP, Finance, Business Intelligence, Web Applications, Property Management, Office applications…

All of BEWANs applications have been developed in-house and can be easily adapted in order to fulfil specific requirements from customers. However, those (licensed-) applications are not cloud-ready, not SaaS-ready and therefore run on private machines.

A project can be:

- A turn-key project: in this case BEWAN takes care of the whole IT project: consulting, requirements study, hardware implementation, software development & implementation, network, training, helpdesk & support, and all related services.
- An integration project: in this case BEWAN integrates software modules within an existing infrastructure or an existing main application.
- An ICT project: in this case BEWAN takes care of the infrastructure project by means of selling hardware, consulting & support services for servers, workstations, networking, hosting, security, online backups etc.

From BEWANs point of view, the 'solution' is far more important than the 'product'. In our vision, the following factors are important and strategic differentiators:

- Knowledge of the business of our customer
- Knowledge of the customer's needs
- Understanding of the customer's vision and strategy
- Knowledge of the modern ICT technology in order to be able to right-size the platform on which the solution will run
- A well-structured, but 'human' approach in developing, implementing and servicing solutions

These 5 pillars lead to:

- Long-term Solutions & Services that not only reach a high level of functionality, usability, efficiency, technical stability, scalability, maintainability and extensibility, but also support the management in measuring performance, by means of put-forward Integrated Business Intelligence tools

During 2011/2012 BEWAN took its first steps in cloud computing by means of developing its own cloud infrastructure based on the OpenStack cloud operating system. Until now, this infrastructure is limited to IaaS and is used to host some internal applications and a few websites ("private cloud").

As already mentioned, BEWAN is now in a project of redevelopment and this time BEWAN wants to hit the ball right and take advantage of what PaaS in general will offer.

4.2 Objectives

BEWAN is in a process of redeveloping its standard applications and the objective is to propose SaaS – multi tenant software solutions in the cloud. Depending on the usage and load, BEWANs objective is to be able to deploy applications to its private cloud and also to be able to scale out to high performance public clouds when needed. Off course those private and public clouds should offer the services needed by the BEWAN applications.

4.3 Current Status (as-is)

Over the years, BEWAN has used different technologies to develop its standard applications; also BEWAN still maintains some applications which were built by a company which was acquired by BEWAN in December 2010. Some of the technologies used include: Uniface, UNIMS/4GL, Delphi, Magic, C#, VB, .NET, PHP, Zend Framework, Doctrine ORM... on Windows/Unix/Linux. Most of BEWANs applications use MS-SQL Server, Sybase Adaptive Server, or MySQL as DBMS.

It's clear that all those different technologies, in the long term, are an issue, considering their support and maintenance. Adding functionality, changing applications due to new legal, tax or other regulations and so on ... involves that many applications have to be modified. This is quite time and money consuming.

4.4 Target Picture (to-be)

BEWAN already started to tackle the issues by pointing out a single architecture for our next generation products. The new software architecture is based on SOA principles and applications-services are being developed in C# for the .NET framework. Visual Studio, is used, with extensions for WPF (presentation layer), Workflow Foundation, ORM (NHibernate), Service Bus (NServiceBus) and Team work (Team Foundation).

Because of the SOA, all components are loosely coupled and an application can be build by composing/connecting UI's, business processes and underlying services (business services, persistency services, reporting & BI services) . However, looking at deployment and execution, there are still some questions and the expectation is that PaaSage methods to simplify the life of the developer. The base line of PaaSage: “Develop once, Deploy many” is very appealing to BEWAN, and PaaSage should put the developer in a position where he/she doesn’t have to worry about deployment and execution.

In the project, web UI’s, tablet UI’s are also foreseen. There is a big chance that Eclipse with the PHP extension, maybe later with .NET extensions will be used, but this is not 100% clear for now.

Figure 4-1 illustrates the global architecture.

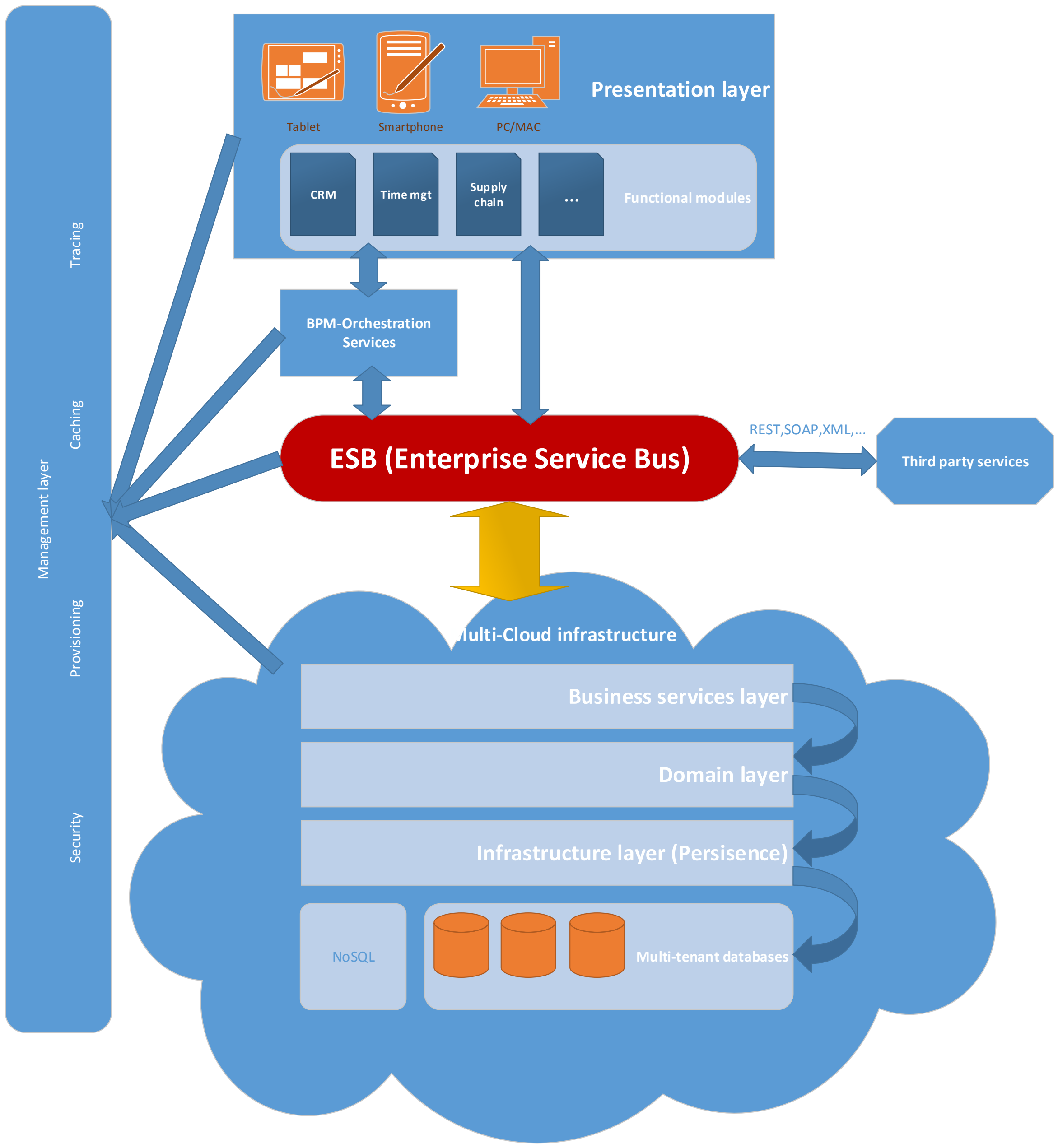


Figure 4-1 global architecture

Figure 4-2 and Figure 4-3 illustrates the application architecture.

Of course, this is really the “big picture” of BEWANs application architecture and too huge to be used as a use case for PaaSage. In the next chapters, one module of the application, the ‘after sales’ module, will be proposed.

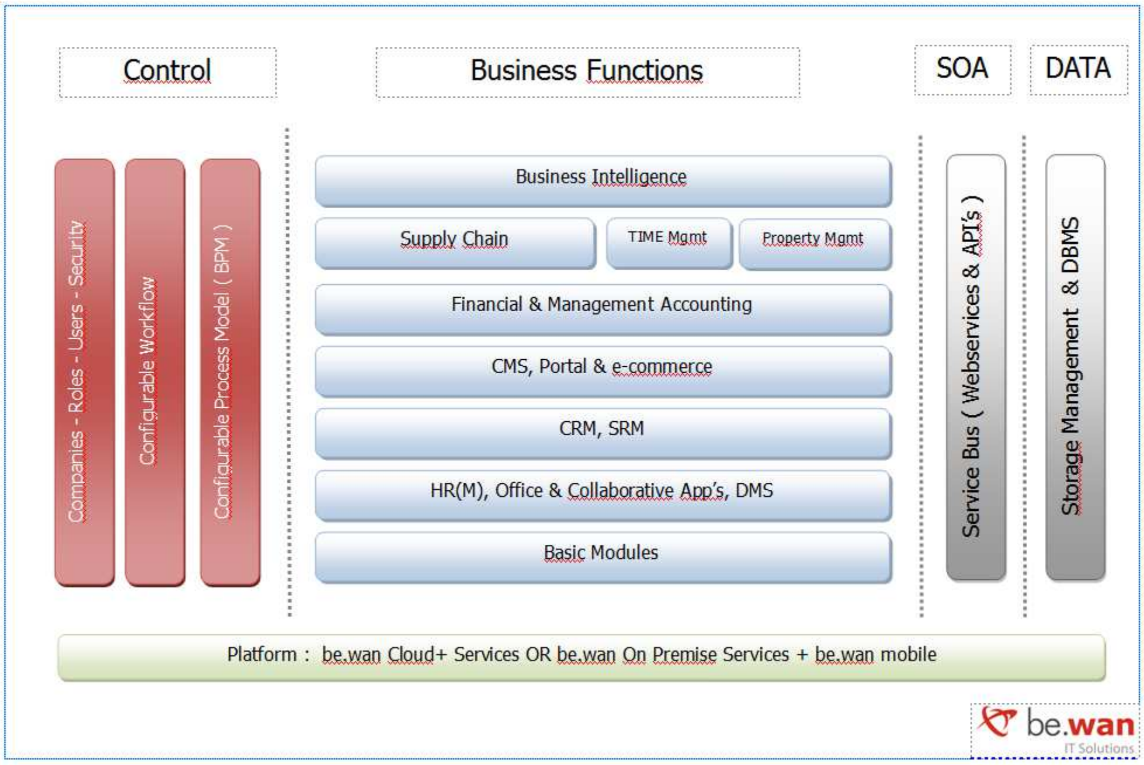

Control
Business Functions
SOA
DATA
Companies - Roles - Users - Security
Configurable Workflow
Configurable Process Model (BPM)
Business Intelligence
Supply Chain
TIME Mgmt
Property Mgmt
Financial & Management Accounting
CMS, Portal & e-commerce
CRM, SRM
HR(M), Office & Collaborative App's, DMS
Basic Modules
Service Bus (Webservices & API's)
Storage Management & DBMS
Platform : be.wan Cloud+ Services OR be.wan On Premise Services + be.wan mobile
be.wan
IT Solutions

Figure 4-2 Overall application architecture

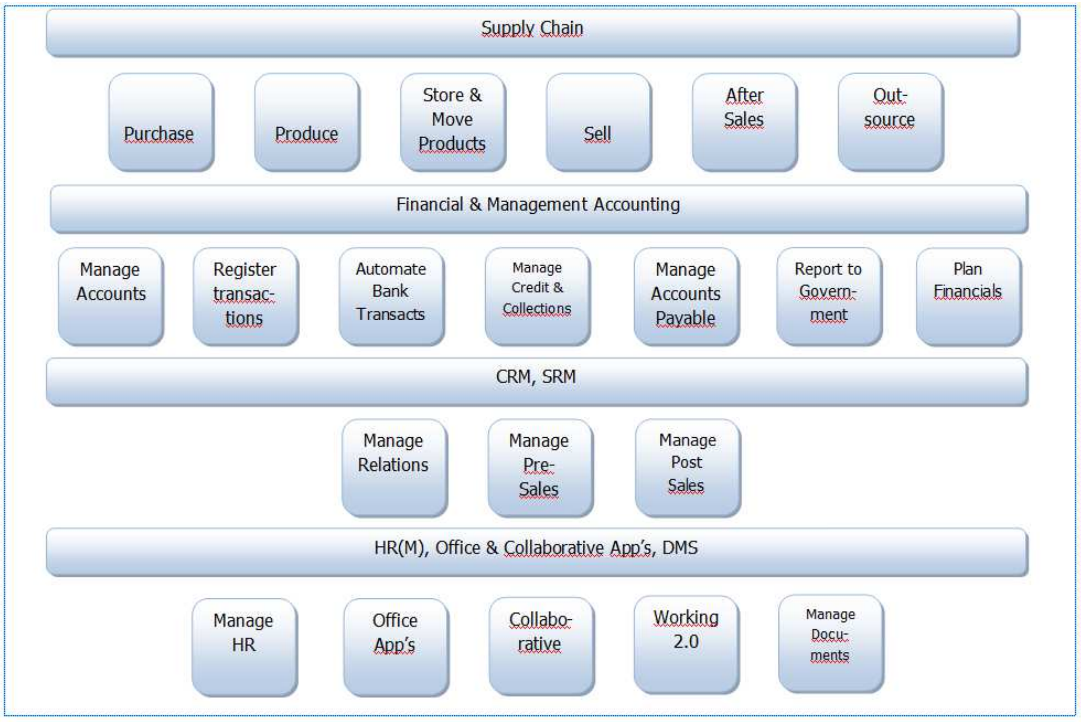


Figure 4-3 Extract of functional architecture

4.5 *Objectives 'After Sales' module*

The objective of the use case is to develop an application which supports the process of the after sales department, in a way that the activities of the different actors in the process are better aligned and real-time integrated. BEWAN wants to offer this application as a cloud application in a SaaS model.

4.6 *Current Status (as-is)*

Many of BEWANs customers have an after-sales department which takes care of the on-site installation, maintenance and repair of the products (machines and equipment) that they sell. Some of those companies also out-source this activity to a specialized company; however this business model is left out of scope.

In many cases, there is still a lot of manual work in the after sales department. Most of the activities are followed up by human actors who communicate most of the time by phone (i.e. planning and work order distribution) and where data is distributed and collected by paper (i.e. work-orders, service reports). This leads to inefficiency, errors, stock-levels for parts not up to date, multiple interventions for the same repair order, late and wrong invoicing because of the manual work etc.

Let us look at a "high level" workflow in order to better understand the process.

- Start events :
    - A customer requests a service (repair our maintenance)
    - The maintenance application generates a recurrent maintenance task
    - The sales application generates an new installation request
- The planner prepares the work-order and plans execution for a technician on a certain date.
    - Reservation of standard boxes containing the parts for a standard maintenance for a given machine
    - Prepare all the parts for a new installation and print a delivery note
    - Prepare or order parts for special repairs or maintenance
- Technician comes from time to time to the dispatching to take his work orders/delivery notes and picks up the material in his truck (standard boxes and/or special parts, new machines to be installed)
- Technician goes on site and carries out the task and writes his report : which and how many parts he used, how many time he spent and other observations
    - It also can happen that a part is not available in his truck, in this case the technician calls dispatching to order the part, the task is then suspended and rescheduled when the part becomes available
- Customer signs the report, gets a copy and agrees that the job has been completed
- Technician comes back to the dispatching and returns his reports, the boxes and unused parts
- Dispatching enters spent time, used and unused parts based on the hand-written report
- Dispatching refills the standard boxes and enters the used material in the inventory system (change of stock location)
- Sales admin makes the invoice for the customer or makes a claim to the manufacturer in case there is a warranty

Looking at this “simplified” workflow, it’s clear that there is still manual work that can be automated, that different actors and systems are involved in the process and that there are delays in the process which make that data in back end systems are not up to date. Also, the fact that hand-written reports have to be made by the technician and that these reports have to be re-entered is a big source of errors and loss of time.

4.7 Target Picture (to-be)

According to the above workflow, the problem is that at a certain moment, the technician leaves the company’s premises, travelling to the customer, and is thus disconnected from the automated process. Everything that happens after that is not under real-time control and has to be written down on paper, communicated by phone, and so on in order to be re-entered into the system once the technician comes back to the dispatching. Consider also that technicians do not come back to the dispatching every day, in most cases they come only once per week.

So the target is to develop a mobile or web application which is synchronized with the back office applications avoiding paperwork and re-entering data. The main functions of the new application should be :

- Server side (running in the cloud) :
    - Offer services to receive master data (customers, contacts, install base, technicians, documentation) from the back office applications
    - Offer services to receive the planned service tasks with detailed list of the parts and the boxes from the back office applications (planning and material lists)
    - Offer services to the technicians client application to get the tasks
    - Offer services to receive service report data from the client application
    - Send the service report (in pdf format) to customers and to the back office
    - Offer services to send back service report data to the back office applications (for automatic processing of used parts, spent time and invoicing)
- Client Side (running on a mobile device)
    - Get the service tasks
    - Enter service report (spent time, used parts, other observations)
    - Close a service task and accept a signature by the customer
    - Send service report data to the server

The application also has to provide useful functionality to the technician such as :

- Agenda and query open tasks
- Contact information
- Email and Instant Messaging
- Customer history
- Machine history
- Documentation on the installation (i.e. electrical schema)

4.8 Development and deployment

As the application will be used by many companies to support their after sales department, users using devices running on different operating systems on the client side will be encountered: laptops on Windows, tablets on iOS, Android, Windows RT, etc.

In some cases, technicians already have a laptop with them because they need it to program or re-program machines, or to read out the event log of the machine.

So the easiest way to fulfil the requirements of such an application would be by developing a pure web application accessible via a web browser and using web services running on the server. However, this means that the user always needs an internet connection on a WiFi or 3G, which is not always possible.

On the other hand, developing a native asynchronous offline application which synchronizes with the server anytime there is an internet connection available is difficult too; because of the different OS on which different devices are running and the delivery models those platforms are using (app stores).

5 Public sector – electronic portal for citizen-city

This case is supported by the EVRY partner.

5.1 Organisation behind Case

EVRY is one of the leading IT companies in the Nordic countries, with a strong local presence in 50 towns and cities. Through its knowledge, solutions and technology, EVRY contributes to the development of the information society of the future. EVRY combines in-depth industry knowledge and technological expertise with a local delivery model and international strength.

EVRY has some 10,000 employees, and the company is committed to demonstrating that Nordic customers are best served by a supplier that understands Nordic business from the inside. EVRY reports annual turnover approaching NOK 13 billion. The company is listed on the Oslo Stock Exchange with the ticker EVRY and operates from headquarters in Oslo, with major activities in both the Norwegian and Swedish markets.

Within EVRY there is an own Business Area (BA) for Public Sector, and within the BA there is a Solutions Service Line which is responsible for Product development & Product Management, Delivery of solutions (Consulting), ASP/hosting of applications and Customer support**.**

5.2 Objectives

The Norwegian Public sector is under pressure to develop more efficient ways of providing services for the inhabitants and businesses of Norway. In the next ten years the demographics of Norway will go through a significant shift where a large proportion of the population will transfer from being of working age into retirement. This will give two effects on the public sector: the demand for public services will increase significantly; and there will be a reduction in the total size of the workforce. ICT will be a significant driver to reduce the negative sides of the demographic change.

There are currently 428 municipalities and 19 regional mid-level governmental districts in Norway today. Municipalities have the responsibility for several services to the local community such as nurseries, schools, infrastructure, and regulations for properties and real estate, and providing services for the business sector. For administrating these services, they use various applications and solutions, for example workflow management tools.

Most municipalities host their own applications locally, or in cooperation with directly neighbouring municipalities. To meet the challenges of the future with more efficient and Citizen-centric solutions there is a need to renew and rethink how current applications are used. In a typical case such as managing a request for building a house, municipalities also need to take advantage of external services such as databases on housing regulations or detailed maps of the area, and integrate these within their processes. The integration should be between the processes as well as archives. Software provided to municipalities can be remote or in premises, or ASP vs. SaaS, thus several delivery models need to work together.

As a major ICT vendor in the Nordic Market EVRY wants to position itself for future business models. We want to use our existing applications with a large local installed base in the municipal sector and integrate these with a cloud offering delivering standardized citizen processes where the process is run in the cloud, but closely integrated with the business applications installed at each individual customer.

The public sector use case will establish how CloudML can be used to develop services that encompass locally installed applications that get value added functions delivered on a cloud model. For example to support a process where a workflow and User Interface is run in a private cloud, but it reuse public data/Open Data-databases, and integrated with locally installed archiving and accounting systems for a municipality. For the Citizen the solution appears seamless across services in the private cloud, public cloud and on locally installed platforms.

5.3 Current Status (as-is)

Municipalities have the responsibility of offering public services to citizens while they vary on size and processes and therefore the complexity of the problems to solve. Many municipalities are still working with systems developed in the 1990s while they see the potential of savings by sharing services and processes with other municipalities. The common challenges in this trend are to provide trust-worthy, reliable, and accessible services to the citizens while, at the same time, respecting the municipalities' limited access to qualified staff and limited budgets. Improved efficiency, sharing of resources and integration between processes are major means to cut the costs. Innovation, self-service, optimization, standardisation, security, flexibility, and economical predictability are the benefits.

EVRY develops and delivers Off the shelf (OTS) solutions for case management and archiving (ephorte and ESA), ERP-systems (System 4 Agresso), Web-portals and intranets (Interaktor, built on Microsoft SharePoint), and centralized solutions for open data/public and private structured information (InfoTorg containing Credit information, address registries, citizen register, building registry, Cars and boats registry, enterprise information and more).

There are two potential use cases that are relevant for use within the PaaSage project.

- Parts of process for application for building permits
- Time management and internal HR process

Use case for Building permits

In Norway (as most European countries) a landowner needs a public permit to set up a new building and the municipality need to approve of the architecture, lay-out etc. This process consists of several steps defined in the Norwegian legislation.

In all municipalities today, handling building permits is a very manual process.

The AS-IS process is very manual. The figure below shows how an agent (either Citizen or a Constructor) must proceed to send a notice to neighbours as the first step in a process of getting a building permit. The agent must contact the municipality physically or by mail to get a list of all neighbours to his property. As this list is generated by a look-up in two independent registries it us usually done manually by an employee at the municipality. When the agents received the list of neighbours he/she must either present drawings of the new building to each neighbour physically, or send these as registered mail. Registered mail is costly and requires the recipient to physically visit the post-office for identification to receive the letter.

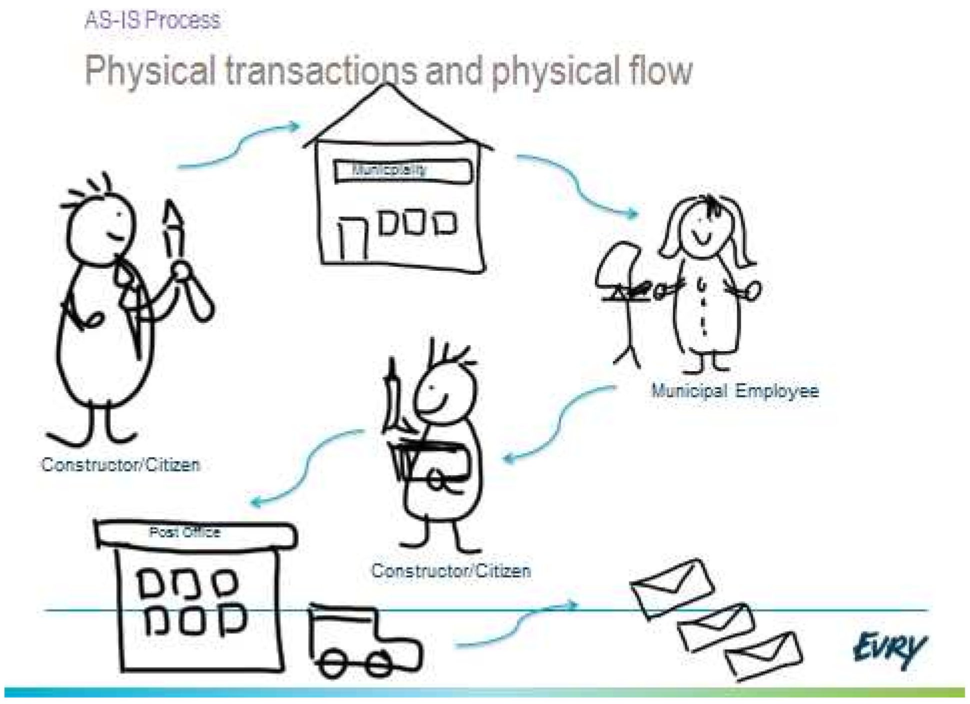


Figure 5-1 Physical transactions and physical flow for building permits

When an application is sent to the municipality the majority of applications are manually registered in the case management system.

There is obviously an opportunity to reduce costs related to both workflow and availability. All Municipalities must follow the same laws and regulations for their core services processes. A build once, deploy in several organisations will provide a low cost option for distribution of best practice services. Internet-based workflows will also reduce travel costs as citizens can be served online rather than in physical transactions. Introducing cloud based delivery models can improve ease of operations for the individual municipality as they can reduce the internal resources spent on hosting, operations and maintenance of locally installed software.

A cloud-delivery model also provides scalability across organisations, and across processes that have unsynchronized peak-periods over the year. Cloud delivery models are assumed to provide more efficient hardware-usage compared to multiple local servers with similar software.

5.4 Target Picture (to-be)

In the very long term (10+ years) we believe most municipal IT solutions will be hosted in hybrid cloud models with some services in private clouds and some services in public clouds. Services must therefore be able to communicate seamlessly across different cloud-based applications.

Getting to this future state will contain trial and error experiences where applications are gradually shifted from locally installed software to gradually more cloud based models. In this transition solutions for the public sector must be able to maintain locally in-house installed applications, but integrated with cloud services for reuse of data stored in external databases, user interface and business process management, authentication and digital signature among other services.

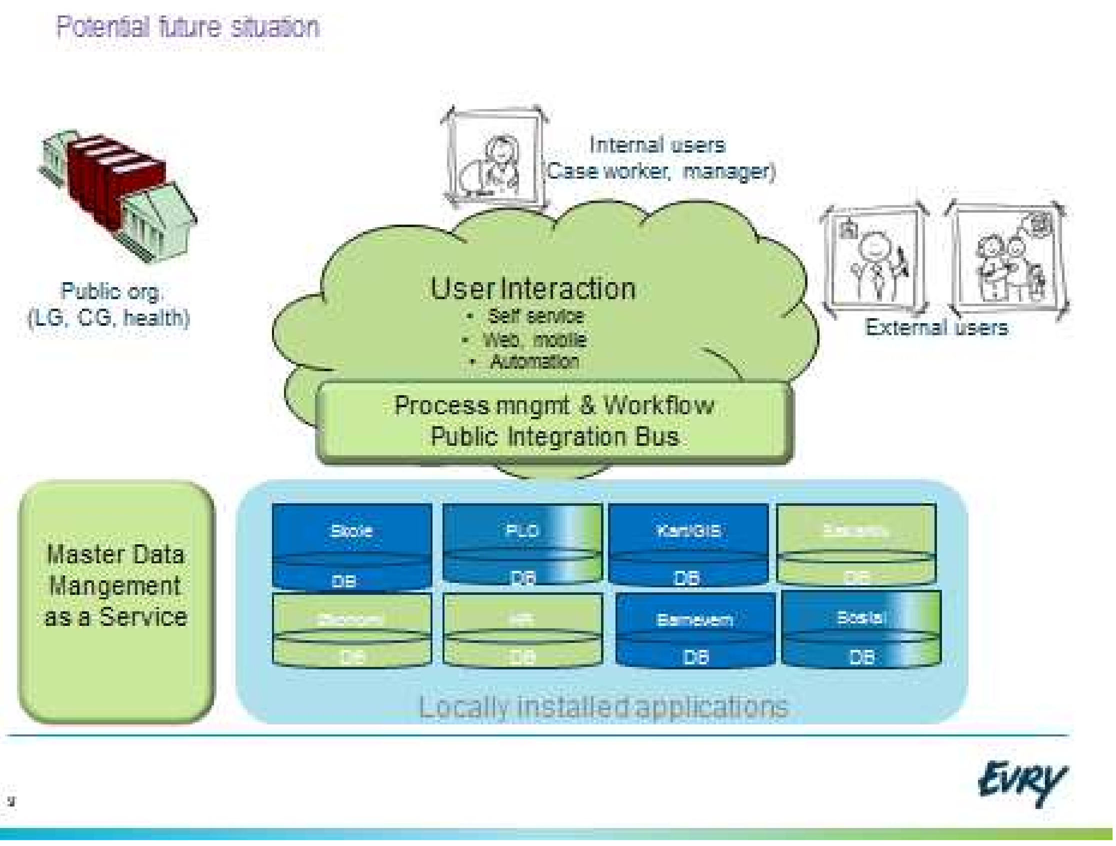


Diagram 5-2 Vision of the future deployment of eGovernment in Norway.

CloudML has the potential to be a major differentiator in this transition, and to enable an efficient “end-state”. EVRY has chosen to start our journey into cloud based offerings to the public sector based on a Platform as a Service solution built on Apprenda .Net. EVRY intends to provide master data management of public data (citizen registry, real estate registry, car registry etc.) as a service, provide user interface, Enterprise service bus and integration as a service, but closely integrated with locally installed applications in the individual municipality.

Use Case building permits.

To increase efficiency in the building permits process EVRY wants to use a cloud based application. This cloud based application will consist of primarily WebGUI, workflow/rules engine and integration bus, and it will reuse data available in external databases or locally installed application at the individual municipality.

The diagram below shows how EVRY expects the To-Be situation: The agent logs into an online service – ServiceBox. He/She can perform a look-up of all affected neighbours (combined register look-ups), upload the relevant drawings and documentation and the notice is electronically distributed to each of the affected neighbours.

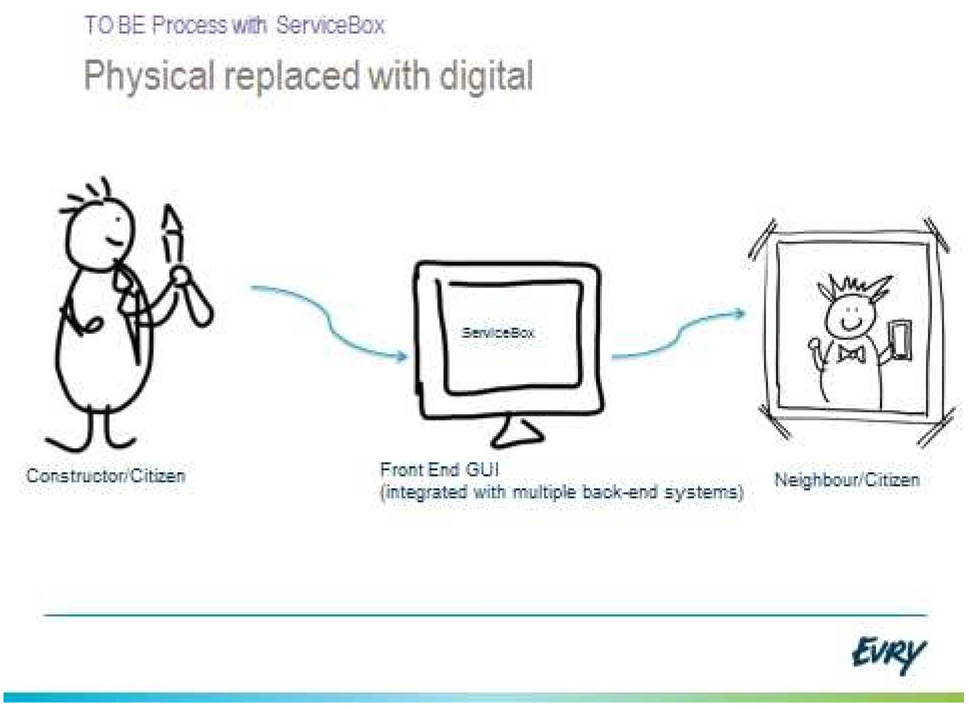

TO BE Process with ServiceBox
Physical replaced with digital
ServiceBox
Constructor/Citizen
Front End GUI
(integrated with multiple back-end systems)
Neighbour/Citizen
EVRY

Diagram 5-3 Vision of the ServiceBox for going digital interaction with the citizen.

Future dialogue between the municipality and constructor is done electronically through the solution. At the end of the process there is integration with the municipal accounting and invoicing system so that the constructor is invoiced the application fee.

EVRY business needs/requirements toward PaaSage and Cloud ML:

- Reduce technology dependencies/platform lock in
- Integration across cloud and a wide range of local applications
- Framework for modelling and analysing legacy and cloud applications in order to understand their delivery models and services and find integration solutions
- Framework for «SOA/Cloud» Governance to keep control on dependencies
- Scalability (across data centres, and across business processes over the year) and “portability” between data centres

User-organisation «business needs»

- CloudML to ensure good modelling processes and dataflow across cloud and local solutions
- Metadata for efficient reporting on
- End to End security and data integrity

5.5 Walkthrough PaaSage Workflow

In this section a case-specific walk through of the business needs in the Public Sector Use Case. Due to the uncertainties of how the specific case will be implemented with the PaaSage technology the Focus of this description are the business needs of EVRY and how we will meet our customer requirements. A breakdown according to the PaaSage workflow is not been considered appropriate to describe the business needs of the Use Case.

5.5.1 Overview of architecture

When a user attempts to use a service, he/she goes through the firewall and meets an Access Manager. Access Manager decides whether the user has necessary authentication information (credentials). If not, it dispatches the user to Authentication Provider. We plan on using ID-Porten, a Norwegian government authentication provider for national electronic IDs. Having successfully signed in, the user is dispatched to an Enterprise Service Bus (ESB) where all the services reside.

ESB is primarily a proxy for services. It has additional responsibilities like authorization, provisioning etc. Services call each other *preferably* using ESB.

The *state* is communicated among various services using an event (news). A service publishes an event (news) using *Event Write Service*. News Writes Service *may* make a set of operations before it publishes either the original news or a set of new news to *News Server (Atom Feed)*. Other services will each read (poll) events (news) by calling *Event Read Service*.

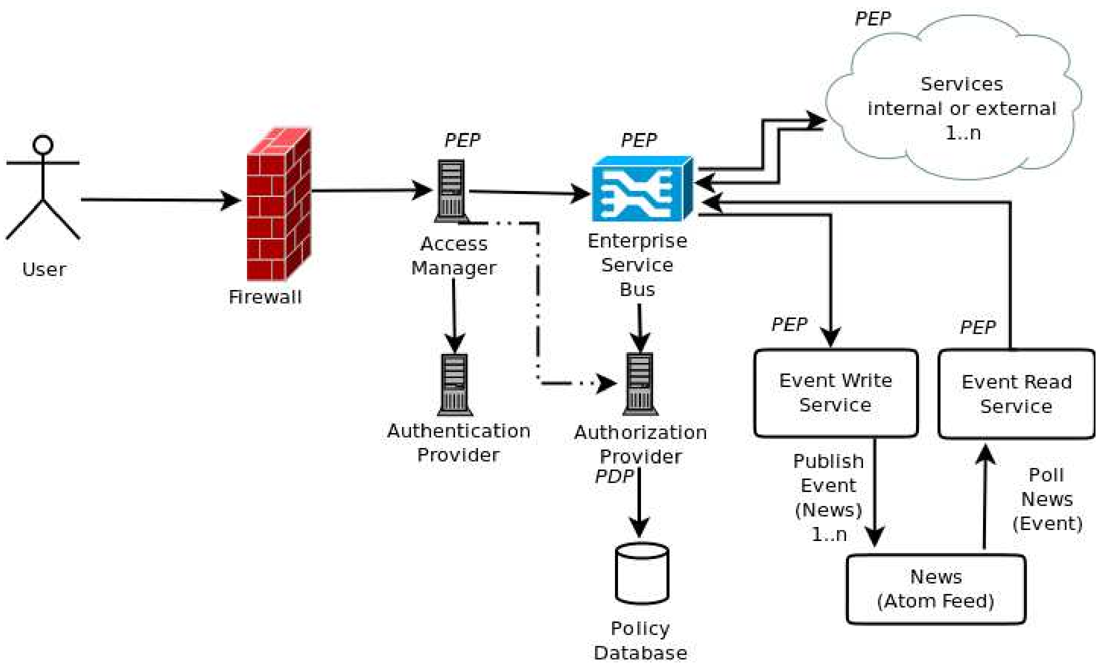


Figure 5-4 depicts a high level view of ServiceBox.

5.5.2 Actors – user types

Primarily, ServiceBox will have two types of actors (users): internal and external. Internal and external users will typically have different access control levels (ACL). Both external and internal users can be either human users or machine users.

Internal users are typically employees of an organisation. Consultants, contractors etc. may also be defined as internal users.

External users are typically civilians who use self services. Note that several special user types may be defined. For instance, a *guardian* is someone who uses some service on behalf of someone else. Enterprise User is a person or set of persons who act on behalf of an organisation.

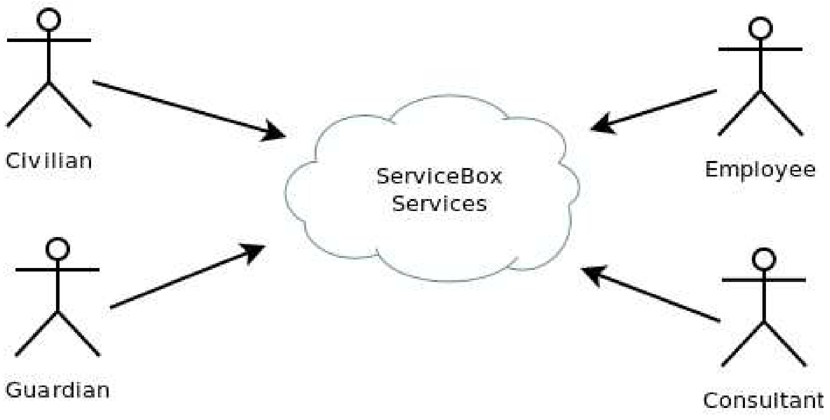


Figure 5-5 shows a partial list of actors. A user may be a person or a service/system.

Note that a user may be another service, sometimes called a system user.

5.5.3 Services

Strictly speaking, services are per definition stateless. Under no circumstance, state should be stored in a service. Stateless service design allows parallel servers and subsequently high scalability and stability.

In our model, services are categorized as:

1. Informational: the service makes only read operations on its back-end systems including the database
2. Read-Write: the service may read and write information on its back-end systems including the database
3. Computational: the service receives input, makes some calculation and returns an output. The service makes no whatsoever read or write operation. Such services are typically CPU or RAM intensive. On the other hand, the service consumes little network bandwidth.

Services may additionally be categorized as standalone or composite (aggregated). A composite (aggregated) service may depend on several other services.

The above categorizations and the statelessness of services impose guidelines for how services should logically and physically be defined. E.g. computational services may be run on multiple computers with some certain load balancing algorithm (e.g. simple round robin).

Public sector organisations reportedly require the services to be formed according to following SOA patterns: http://www.soapatterns.org

Services may be either internal or external. Internal services are developed and governed by ServiceBox. External services are owned by some external entity and ServiceBox has no whatsoever control over them.

5.5.4 Use-Case view

Here, we select a few important steps that illustrate significant aspects of the architecture.

The Basic Use-case described below is parts of a full process involving a Citizen logging on to a public service, apply for a building permit, the application is processed, documents archived and the end user invoiced fees for the process.

A User logs onto the cloud solution (ServiceBox) through ID-porten, a public sector authentication service for electronic IDs. Depending on the authorizations the users is given access to one or several services

The Citizen starts a service "Application for building permit" and opens a form to fill inn and send a notice to both relevant neighbours and to the municipality. The ServiceBox application uses the Citizen ID to fetch data on the user from public data registries (InfoTorg): This information is case specific, but could be:

> *Address, other family members,*
>
> *Tax/revenue information,*
>
> *Ownership of real estate (including GIS information on location, borders, buildings on land etc.)*
>
> *Neighbouring properties and owner of these*
>
> *Ownership of cars*
>
> *Credit information,*

After filling in the blanks, probably uploading files/data from the local PC the User finishes and sends the form. The form-data is sent to both the local archiving system, and into a BPM-flow for further follow-up. In a building process, the follow-up will include

5.5.5 Requirements from EVRY related to the PaaSage workflow/ components

New and legacy applications

- EVRY does not consider it achievable to have profiling of our existing legacy applications. These are too complex to deem it possible for a profiling of applications to cloud-enable them
- New Applications will predominantly be built in .Net programming language.

Cloud ML application model

- This component must support how to model dataflow, users, standards for defining parameters and architecture to comply with PaaSage components
- Must ensure authenticity of data end-to end
- PaaSage must provide models that includes integration across cloud and a wide range of local (i.e. at customer) applications
- Framework for «SOA/Cloud» Governance to keep control on dependencies

Speculative Profiler

None identified

Intelligent Reasoner

Support methods to ensure access control across data sources

Extra functional adaptation

- Must support integration between applications/solutions for both “fetch and deliver” data
- Support methods to ensure access control across data sources
- Support use in a .Net environment

Platform specific mapping

- Reduce technology dependencies/platform lock In. EVRY has chosen Apprenda as PaaS platforms for our developments towards Public sector.
- Must support scalability (across data centres, and across processes over the year) and port between data centres

Execution monitoring

It must enable the monitoring of the responsiveness of different services, both application specific, and responsiveness of external/locally installed systems that are integrated with cloud applications.

Execution Control

It must enable change of application behaviour to enable scalability between data centres.

6 eScience sector – resource intensive simulations

6.1 Organisation behind the case

6.1.1 High Performance Computing Center Stuttgart (HLRS)

The High Performance Computing Centre (HLRS) is a research and service institution affiliated to the University of Stuttgart. It has been the first national supercomputing centre in Germany and HLRS is offering HPC resources to academic users and industry. HLRS provides also consultancy services and training for industry and academia to program large-scale systems and to convert existing applications and algorithms into large-scale use cases for performing the scientific experiments. HLRS work focus is oriented towards:

- Provision of several different high performance cluster systems, allowing the support of differing needs. It is able to provide “thousands of cores” supercomputing capacity.
- Operation of supercomputers owned by HWW GmbH, a public-private partnership consortium comprising among others the German Telecom and Porsche AG.
- Collaborative research with automotive industry goes through the ***Automotive Simulation Center Stuttgart (ASCS)***.
- Services and consultancy for scientific and industrial users.
- Research in the area of supercomputer architectures, simulation software, software engineering, as well as distributed computing and networking.
- Teaching in distributed systems, software engineering and programming models.
- Cooperation with international partners from industry and research.

HLRS is primarily academic (central service institute by the university) but has also industrial users. Industry and academia represent different economic factors (different available budget, different costs per core/hour), but also different stakeholders in HLRS (ownership of resources). Furthermore different requirements in particular in terms of security exist. There are several opportunities for the users to get access to the available systems and run their application:

- Research access to the national supercomputers through review procedure**:** Eligible are applications from publicly funded academic and research institutions in Germany that might have project partners from Europe. Allocations free of costs for projects **a)** requiring 40 million core hours or more by answering the "Call for Large-Scale Projects" [18] of the Gauss Centre for Supercomputing [19] **b)** requesting less than 40 million core hours after submission of a proposal and a review procedure through the HLRS Steering Committee.
- Research access using the PRACE infrastructure: PRACE is a persistent pan-European Research Infrastructure (RI) providing leading High Performance Computing (HPC) resources. Scientists and researchers from around the world can apply for access to PRACE resources (HLRS and further European HPC systems) through a rigorous peer review process. Industrial users can apply if they have their head offices or substantial R&D activity in Europe [20].
- All other users have to explicitly “buy” core hours e.g. through HWW.

As shown above, resources are offered to different use cases, i.e. users compete over the resources – for PaaSage this means that the resources are not by default “available on demand”. Instead, the requests will be queued. Similar to EGI [21], available HPC resources of HLRS can be requested through different channels depending on country of origin. The goal of HLRS is to allow easier access to resources, both in terms of usability and administrative overhead as well as better resource utilisation and distribution of load across PRACE. One of our objectives in PaaSage, the detailed description of which could be found in next section, is to connect multiple HPC systems via cloud for the parallel execution of parameter sweep simulations, which helps to achieve the desired higher resource utilization level as well as better resource usability, and further reduce the administrative overhead for simulation users by providing “Simulation as a Service” in cloud.

6.1.2 Automotive Simulation Center Stuttgart (ASCS)

The ASCS fosters application-oriented research in the field of automotive engineering by the use of information and communication technologies. It also promotes and accelerates the transfer of the latest results of scientific research on numerical simulation. The goal of the ASCS is to provide industry with HPC simulation methods which satisfy high scientific standards and also fulfil ambitious industrial demands. For this purpose and to develop new strategies for the reduction of CO_2 emissions, the association conducts self-selected research and development projects and contract research. In the context of funded projects it is possible to apply to the steering committee of the HLRS for the free use of additional HPC-resources.

The ASCS activities include:

- Conception and implementation of research projects for the development of process-oriented models and numerical simulation methods for the solution of interdisciplinary technical issues, especially if they place high demands on computing power.
- Conflation of forces engaged in research with industrial practice for the purpose of reciprocal exchange on current issues, the dissemination of scientific results relating to modelling and simulation, to be used in practical applications including the method-oriented support of users.
- Advancement of research in the field of high-performance computing and its applications as well as the dissemination of related scientific results.
- Assumption of the automobile-related functions of the generated projects in the fields of simulation, verification and validation, therefore ensuring the industrial implementation of the developed simulation methods by the ASCS as research facility with its bundled know-how derived from members of industry and science as well as its own employees.

The ASCS creates for its members new opportunities to improve CAE simulation methods, e.g. for the optimization of CO_2 emissions or the reduction of fuel consumption or noise, for future vehicle concepts, especially if they place high demands on computing power. The objective is to reduce the time intervals between the definition of specifications and industrial application by combining the expertise from science and industry. Since development time directly relates to costs it is the goal of every car manufacturer to increase time efficiency. The next sections will describe how PaaSage could help within the virtual car development process. This is illustrated exemplarily on the basis of the development of a side mirror.

6.2 Objectives

High performance computing plays an incomparable role in industrial areas and academic researches, particularly for compute intensive applications. As described above, as one of the three largest HPC centres in Germany, HLRS is offering HPC resources as well as consultation for development of and conversion into large-scale applications to industrial and mainly academic users. In particular, one of the major research fields of HLRS is computationally intensive science that is carried out in HPC environments (eScience) including molecular dynamics simulation and biomechanical simulation e.g. blood flow, bones and bone-implant-systems. HLRS is particularly pushing the aspect of convergence between high-end and low-end programming, to enable common developers to exploit new resource infrastructures that scale both vertically (HPC) and horizontally (cloud). Furthermore HLRS is cooperating with ASCS to perform scientific research on numerical simulation in the field of automotive engineering.

However scientific computing requires an ever-increasing number of heterogeneous resources to deliver results for growing problem sizes in a reasonable timeframe and with the current business procedure of HPC, it is difficult for users to access and manage the execution of such applications, in particular applications that involve parameter sweeps, as elaborated in detail in next section. With the recent cloud hype, there has been a growing interest from the eScience and HPC community to exploit cloud infrastructure, as they seem to offer just the capabilities required by the researchers because of its well-known advantages:

1. Strong computing resources (Scalability)
2. “on-demand resources” (Elasticity)
3. High availability
4. High reliability,
5. Large data scope
6. Reduced capital expenditure (cheap).

A lot of research has been done in order to investigate the requirements [3][5][6][7][8][9][12][13] as well as the performance and cost of porting eScientific and HPC applications to different cloud infrastructure [1][2][4][10][11]. The studies have shown that current cloud computing services are insufficient for large scale scientific computing, the performance gap is seen not only in the MPI performance of distributed-memory parallel programs but also in the single compute node OpenMP performance for shared-memory parallel programs in cloud. However, cloud still appeals to the scientists that need resources immediately and temporarily [1]. Scientific applications with minimal communication and I/O are also best suited for clouds. Thus, the HPC community would benefit mostly from a combination of the strength of the two environments.

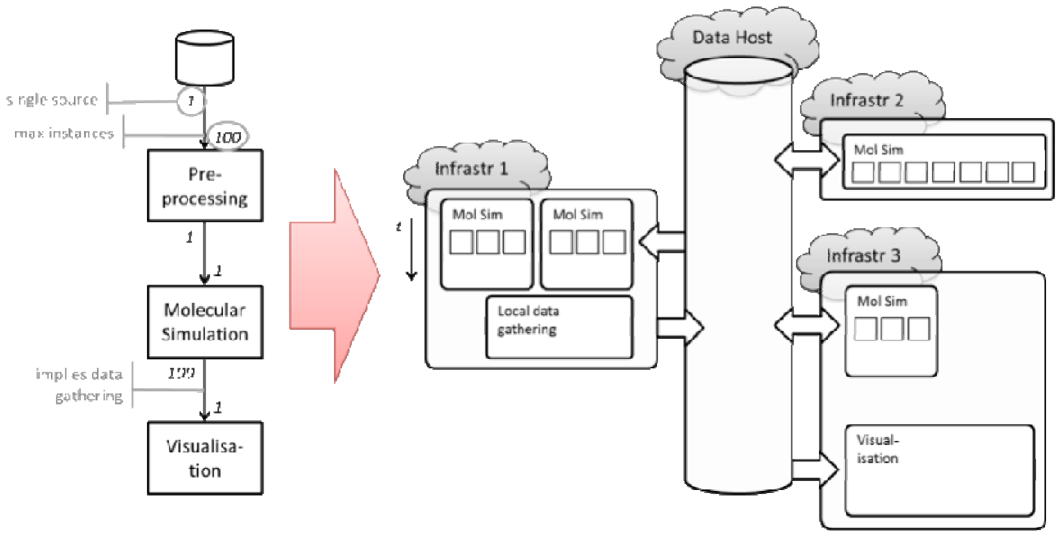


Figure 6-1: Envisioned Execution of large-scale eScientific Workflow across HPC and cloud.

To sum up, the main objectives, which should be achieved through the PaaSage project for facilitating the execution of large scale and heterogeneous-resource demanding simulation workflows, are listed as below:

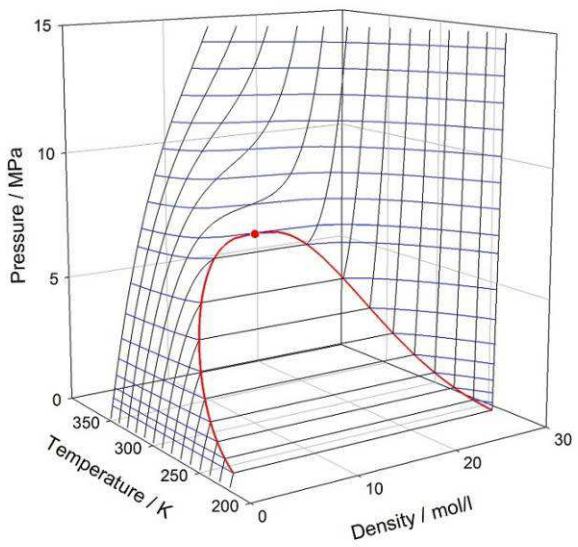
- **Deploy large-scale simulation applications cross HPC and cloud:** Enable execution of large-scale simulation applications across HPC and clouds with a specific focus on molecular dynamics simulation and computational fluid dynamics (CFD) simulation for equation of state calculation. As depicted in Figure 6-1 it will make use of the infrastructure transparent to the user by integrating cloud and HPC environments in a unified model. In other words, use features of both environments, so as to improve performance significantly without increasing the development effort for the user. For instance, in the particular case of eScience applications, this means specifically, that a dynamic amount of parameter sweeps can be calculated over a scaling, compute-intensive simulation. It will implicitly dynamically extend capabilities of High Performance Computing systems with cloud resources to:
    1. Improve the execution of large-scale workflow applications that do not have sufficient HPC resources available.
    2. Minimize the leasing cost while maximizing the contribution to reducing the overall workflow execution.
- **Better resources utilization:** Deploy different modules of the application (e.g. execution and visualisation services) to appropriate resources while taking into account different requirements, e.g. deploy the most high performance demanding parts of the application on clusters and deploy other parts on the cloud to take its advantages like availability, scalability.
- **Expose the simulation application as services:** As the Service-Oriented Architecture (SOA) gains wide popularity, we intend to expose the simulation application as services in cloud that can be easily accessed by researchers and field experts, which allows for multi-tenancy and also sharing of simulation configurations between researchers to reproduce some interesting experiment results. The user thereby gets relieved of the overhead to adjust the infrastructure configuration for the specific use case, respectively vice versa having to develop or configure it for a specific infrastructure, thus avoid many

of the obstacles that currently confound the delivery, accessibility and usability of traditional, non-service-oriented simulation applications.

6.3 Current Status (as-is)

6.3.1 eScience

eScience research is often expressed in terms of large scale computation and/or data intensive science over highly distributed network environments. It depends heavily on the provisioning and availability of computing resources to enable the complex calculations that are part of the respective research fields. Due to the complexity of these calculations, researchers typically rely on strong computational infrastructures; such as high performance compute clusters. However, it can be noted that the underlying algorithm "types" vary strongly, incorporating the full range from sequential, non-scalable programs over embarrassingly parallel instances up to tightly coupled, strongly parallel applications. In addition, in eScience it is often necessary to perform parameter sweeps, which will be elaborated in the selected case of eScience applications below, over the same algorithm in order to identify the impact of variances, so that next to the scalability aspects of the individual application, multiple instances will be executed typically in an embarrassingly parallel fashion, thus adding to the resource hunger.



Molecular Dynamic (MD) or Computational Fluid Dynamics (CFD) simulations are highly representative for modern eScience research tasks. This kind of calculations provides information about how a given substance behaves under a given set of physical conditions, e.g. to predict material behaviour for industrial purposes. For example in MD, to create a meaningful chart (cf. image left) of the substance behaviour within this set of conditions, the same calculations have to be executed multiple times by sweeping the parameter values through the parameter range of each boundary condition. There are different mathematical representations for the behaviour of gas and liquid, depending on the accuracy requirements of the study. The principle thereby consists in calculating the interactions between atoms or molecules within a given volume. The process contains usually several iterations of execution, e.g. first iteration for performing coarse granular simulation over selected points in the parameter space, second iteration for performing fine granular simulations around the point that showed remarkable phenomenon in first iteration. Simulations in different granularity have also different requirements on capability of resource.

Such eScience applications/simulations thereby exhibit two major features that can be exploited for their development and execution, i.e. the usage scenario typically involves horizontal and vertical scale, and they often consist of recurring logical/algorithmic elements. The horizontal scale refers to the amount of instances that are relatively low performing and connected through slow network to satisfy e.g.

changing amount of requests, whereas the vertical scalability refers to the size of the instances themselves and thus implicit to the amount of resources (CPU, memory, interconnect) required in order to address the demanded quality criteria. With the latter respect, there are typical base algorithms and libraries that are frequently used in different contexts with varying datasets – eScience infrastructures typically offer these base environments, whilst most cloud providers as yet fail to incorporate these elements into their platform. This equally includes higher-level simulators, such as OpenFoam, as well as basic mathematical libraries, such as the AMD Core Math Library.

Furthermore, in the eScience research, and in particular the domains promoted here, make strong reuse of existing applications and / or modules which expose essential features of the respective application, e.g. by providing mathematical functions and / or visualisation capabilities and similar. Scientists can easily use this approach to define their own, use-case specific adaptations of the overall application execution, e.g. by loading different data, plugging in additional analysis algorithms, using alternative visualizers etc. Most of these modules implicitly specify the resource capability requirements, e.g. mathematical libraries in eScience applications typically have high computational demands and the type of library implies its scalability scope, respectively restrictions.

Due to the involvement of a number and variety of analysis tools and the strong reuse of existing applications in the scientific problem solving, scientific workflows have become fundamental to e-Research and during the past few years a considerable body of work has been done on the use of workflow systems to conduct scientific applications. Scientists can use workflows to easily express multi-step computational tasks by combining various services, applications and modules. As scientific data sets are consumed and generated by the pre- and post-processors and simulation programs, a scientific workflow describes such dependencies and the relation between data, input parameter set and processing steps which can be everything from short serial tasks to very large parallel tasks (MPI for example) surrounded by a large number of small, serial tasks used for pre- and post-processing. From such workflow descriptions along with the knowledge about the specific use case and modules forming the application, the scalability capabilities and hence the requirements towards the code can be derived.

Currently if a user wants to execute a parameter sweep workflow for MD simulation, solving the according set of equations is already a compute intensive task that will take for example around 20 hours on 4 HPC nodes (i.e. 32 cores) for a single set of parameters. In order to acquire enough data to generate the full information set as needed for the accurate prediction purposes given the condition range, between 100 and 1000 of such individual calculations are needed. This means that with limited resource availability (a single small-scale HPC machine), the straightforward calculation would take up to 20,000 hours (roughly 2.25 years). Employing multiple small-scale machines and increasing the scale over more nodes can drastically reduce the whole time consumption. But even by increasing the scalability to say 20 nodes (160 cores) and employing multiple machines, a large number of machines are needed to be reserved in order to drastically reduce the overall execution time.

Given current conditions, it is already difficult for a user to specify the concrete requirements and provide a suitable configuration for his HPC application. Indeed, to configure the according computing resource triggers many characteristics need to be defined, such as how many cores are actually needed, how much memory is required for computation, how should the machines be configured, expected execution time and so on. There is no general strategy to assess the configuration, as it depends on the specific requirements of the according application and even data. Overestimating the needs will occupy unnecessary computing resources thus leading to unnecessary cost; whereas underestimation will lead to unnecessary delays and even loss of results. The second problem is that (a) if a user wants to rent dedicated resources, a large number of machines need to be reserved in order to reduce the overall execution time, this would require that at any time a certain number of machines are available for usage – with classical setups, this would mean that the machines have to be reserved in advance and the number of nodes is fixed. This is not only costly, but also very inflexible, leading to bad resource load. (b) If the application is deployed on public accessible HPC, the jobs have to be put in a job queue. In this case, users compete for the resources and have to wait for uncertain time before their application can be executed.

6.3.2 Automotive Industry

Nowadays, cloud computing is a much discussed topic. Many companies start to integrate the corresponding concepts into their IT strategies - but not at any price. A generally valid definition is currently not available. This shows that this topic leaves much room for interpretation. Cloud computing exists, but didn’t really arrive in the companies. Nevertheless, in the next years cloud computing will have a lasting influence on the company's work – also the automotive industry. Public clouds, however, are unlikely to be of long-term interest for car manufacturers. Many security and risk issues are currently unresolved in external cloud models. The situation appears different for private clouds and tailor-made solutions for the automotive industry.

Which are the key drivers for cloud computing? This question is answered very differently, depending on which area or person is asked. But the cost aspect seems to be one of the most important drivers. Modern IT services need to meet the demands of car manufacturers (OEMs) and suppliers. The benefit lies in a flexible scalability and a better interception of peak loads. Regarding the application-specific services the current focus mainly lies on Infrastructure as a Service (IaaS) – and especially on the intensive use of cloud storage services. This is closely followed by business applications. Also modern means of communication such as video conferencing, document sharing and social media are in the favor of the users. Unfortunately, the usability is strongly limited through the complex infrastructure of the OEMs.

A successful use case of cloud computing is currently in the areas of sales and marketing. However, opportunities for cloud computing could be leveraged by new e-mobility concepts, such as car2go – a car sharing program, or *car2gether* – a ride sharing platform. System services are working together with car manufacturers and other suppliers in order to develop standards and new applications for the connected car of the future. They work together in building a responsible e-mobility market place, at which the suppliers of such services (e.g. energy suppliers or fleet operators) can place and market their offers via cloud computing. Vehicle owners can obtain these services then through the marketplace.

Even for small and medium-sized enterprises (SMEs), the readiness of the users for the “experiment” cloud seems to be limited because of a high trust in in-house data storage. An interesting question is how SMEs are currently using the cloud and what they want to use soon. In fact, at the moment the SMEs only make little use of the cloud services - practically only e-mail services and web hosting.

The research and development (R&D) departments of the OEMs are reserved, too, because they are dealing with a variety of sensitive data. And this is especially the case in the area of "virtual engineering", i.e. the design and functional design of the vehicles in the early development phases (conceptual design). Generally, the traditional vehicle development using real prototypes can be subdivided into 4 stages (see also Figure 6-2 for a crash testing example): sourcing & assembly, set-up, execution, and analysis. The transition from hardware-based to the virtual development requires a consequent and continuous transfer of all 4 stages from the road to the test bench and finally to the computer.

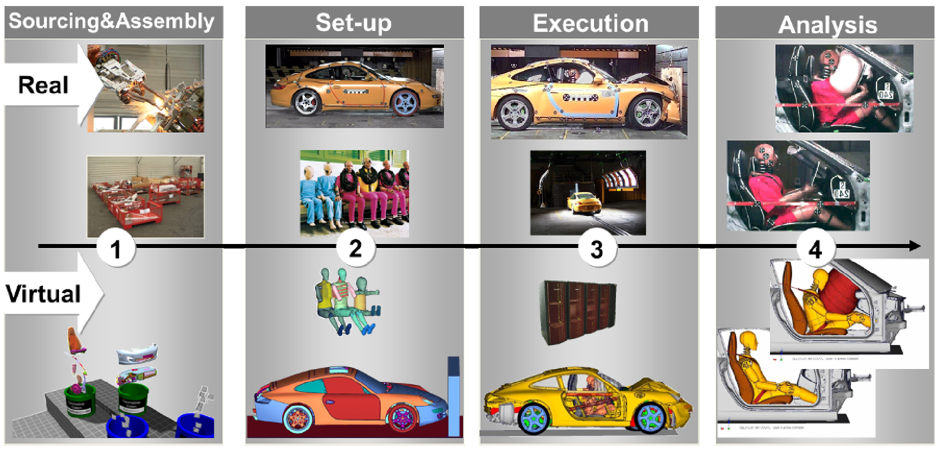


Figure 6-2: Real versus virtual crash testing (source: Dürheimer, Porsche AG, 2008).

In the automotive industry, a remarkable shift from design processes based on physical prototypes to a computationally aided development process based on virtual prototypes is recognizable for the last couple of years. Especially in the concept phase, the most concept relevant decisions are made on the basis of simulation results. In the "Computer Aided Engineering - CAE" in the early development phase, simulations for the fluid dynamical (Computational Fluid Dynamics - CFD) and structural mechanical (Computational Structural Mechanics - CSM) design of the vehicles are carried out intensively. For the functional layout of the vehicle structure with respect to passive safety, usually more than 4000 full vehicle crash simulations are carried out compared to about 150 real hardware crash tests. To optimize the pedestrian protection, even more than 12000 simulations can be necessary.

The typical workflow for the virtual car development process is illustrated in Figure 6-3. Starting with the preparation of CAD models, everything needs to be transferred into meshed models and set up with the respective material, boundary and other solver specific information. As soon as the pre-processing phase is completed, the simulation starts. In contrast to simulation tasks like e.g. for performance, fuel consumption, control strategies, or global heat balance of a vehicle, CFD as well as CSM can't be performed on simple workstations but need to be carried out at supercomputers. Very large CAE models and simulation results must be transmitted (data volumes in the range of Gigabyte) and therefore very powerful networks are required. Besides a very fast network between the used processors (e.g. Infiniband), the performance of the processors among themselves has to be comparable, otherwise the slow processors impede the fast ones. The memory requirements for such CAE simulations are also extremely high (ca. 24 – 36 Gigabyte/processor) and are currently not met by many systems.

On top of that, on the used cloud systems the same CAE software and the same release version of the software must be implemented. The simulation results are subjected to unavoidable variations when they are performed on different hardware platforms.

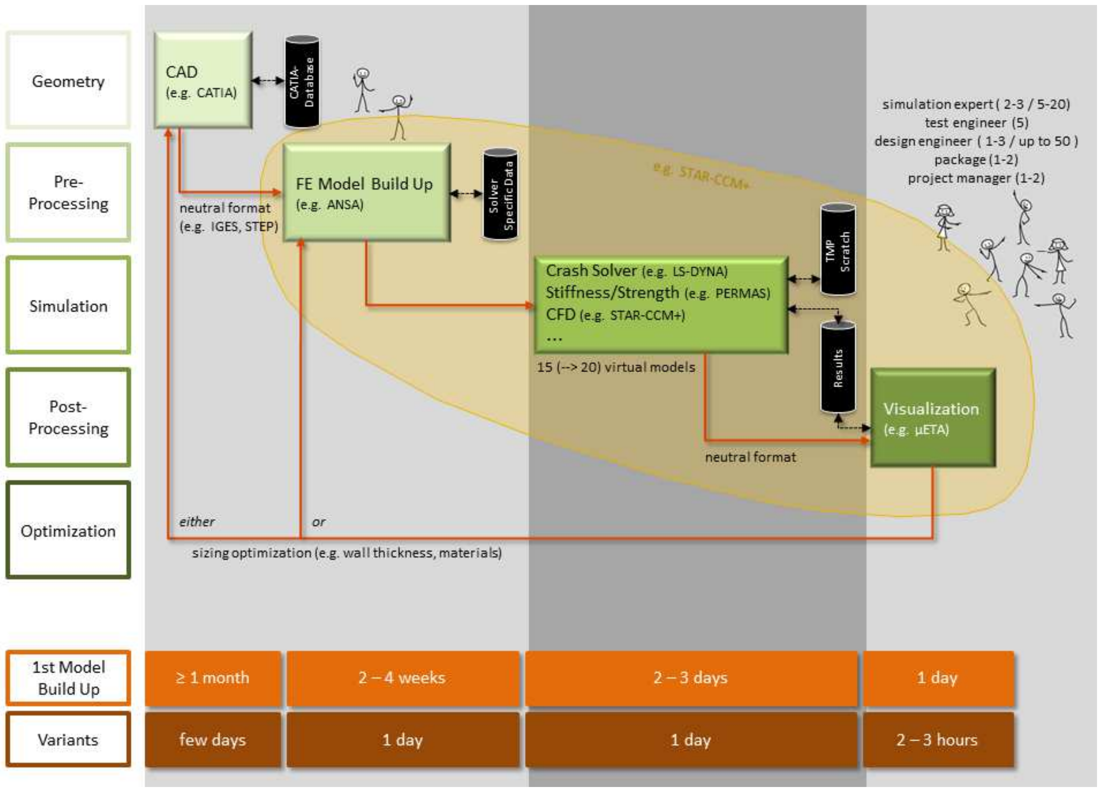


Figure 6-3: Workflow for the virtual car development process.

Finally the simulation results are analyzed and evaluated. Visualization tools are used to discuss the output amongst several institutions. Thus simulation experts sit together with test and design engineers and people from the package or management. All together the consortium can consist of 10-100 participants, strongly depending on whether just a part or even the full vehicle is the object of investigation. Together they decide on possible or necessary changes with regard to e.g. design, wall thickness, materials, package or cost aspects etc. Afterwards another iteration starts, beginning either with changes in the pre-processing phase or even with a modification of the CAD model data.

Another advantage of the virtual vehicle development besides the cost aspect is a reduction of precious development time. Whereas the first loop of pre-processing, simulation and post-processing normally takes a couple of weeks, the following iterations or optimizations can be realized within a few days (see Figure 6-3).

An exemplary use case is the development of a side mirror. The current mirror development process combines both, experimental techniques and simulation methods. Various areas of development are involved, such as styling, engineering, testing, simulation and approval. Basically, the three criteria #1 styling, #2 field of view, and #3 flow behaviour (including impact on fuel consumption and noise emission) need to be taken into account.

The process is as follows. In the early concept phase, several styling designs (5-10) are created, either as plasticine models or virtually. In the very first step of deciding whether a mirror design “stays in the race” or not, corporate philosophy plays a much more important role than the field of view or flow conditions surrounding the mirror. Only the approved design proposals pass through the next stages, namely the field of view and flow analyses. The field of view can be verified with a relatively simple process. On the one hand, various mirror geometries are physically attached to the vehicle and then analyzed and evaluated stationary and during driving. Obviously this is time-consuming and costly. On the other hand, OEMs more and more make use of modern virtual methods. Diverse virtual mirror geometries are instantaneously installed on a virtual driver's seat to perform studies of the content in the mirror and evaluate the visibility. Again some of the mirror designs might be discarded while the others undergo the most expensive/complex part of the development process, the CFD analysis. The traditional way is to perform wind tunnel experiments which require the use of full vehicles. Faster, cheaper and much more flexible is again the virtual counterpart, i.e. flow simulations on HPC machines. The investigated mirror designs are calculated, evaluated, compared to each other or wind tunnel results, and optimized from the flow and pollution point of view. The whole design process is iterative, and most often the best compromise between styling and functionality.

To summarize, the current side mirror development process includes experimental as well as virtual methods and still a lot of manual work regarding the simulations and evaluations of results which could be automated in the future.

6.3.3 Summary

As shown above, in both eScience and automotive industry cases a large number of simulations need to be carried out, output needs to be visualized and several iterations of execution with changing parameters and boundary conditions are required. In summary, the classical issues in high performance computing scenarios we are facing are:

1) Simulation applications like parameter sweep require scalability in both horizontal and vertical direction, i.e. with respect to number of instances and their performance.

2) The number of instances is not necessarily fixed, as the scale may depend on previous results, thus the fixed reservation model of HPC providers is too costly for eScience.
3) HPC resources are typically not available “on demand” – in combination with the reservation issue, it is difficult to “probe” individual parameters before selecting the full set.
4) Existing cloud infrastructures are not adjusted to the specific needs of simulation applications, i.e. they don’t offer the right functionalities, and they cannot support the vertical scale with the necessary performance.
5) Currently, the application needs to be carefully adjusted to individual HPC destination platforms in order to ensure performance.

6.4 Target Picture (to-be)

As shown in the previous sections, the numerical simulation is main research topic for both, HLRS (eScience community) and ASCS (automotive industry), and we are facing same issues when executing multiple simulations in parallel with changing boundary conditions. On one hand, the existing Grid and HPC infrastructures are way too difficult to handle and cannot address the unpredictable dynamic resource need for such kind of applications, which put forward requirements towards both high performance and cloud capabilities for dynamic provisioning of huge amount of heterogeneous resources. Therefore to execute such simulation applications across HPC and clouds is a means to overcome these classical issues.

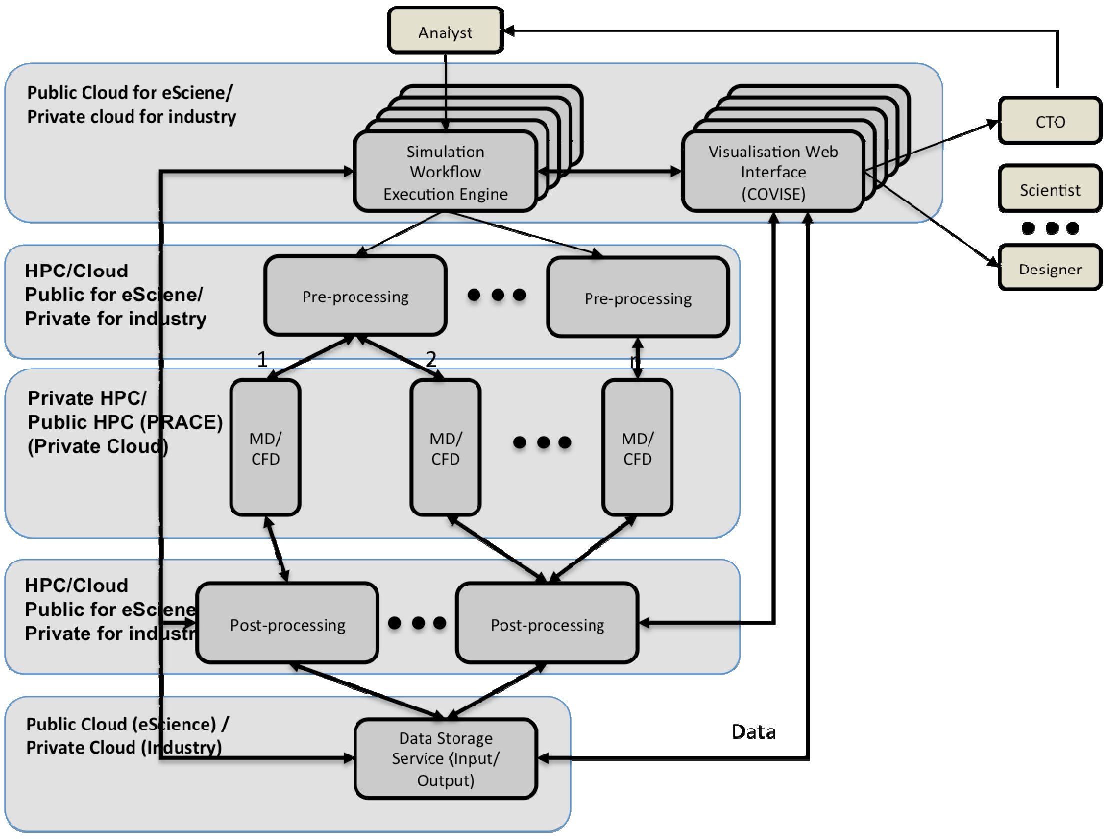


Figure 6-4: Architecture of target application across HPC and cloud.

In order to enable the execution of such simulation applications on various resource environments, PaaSage has to be able to support workflow-like applications such as depicted in Figure 6-4. An approach may involve the following main modules:

Workflow engine that is responsible for the configuration, instantiation, execution, monitor and control of distributed tasks across cloud and HPC. This includes the necessary access rights, data conversion, scaling behaviour, implicit adaptation to the infrastructure and identification of appropriate distributed resources. A prototype of the workflow engine is already available which is implemented in C# by using Microsoft .NET technologies, such as WF [14] and WCF [15]. In this approach a combination of Job Submission Description Language (JSDL) and Windows Workflow Foundation are used to describe a scientific workflow. Both of them are in extensible XML format. The JSDL [16] is specified by the Global Grid Forum for describing all the information needed to run an executable on compute resources including requirements on hardware and software as well as the data dependencies.

Visualization web application is responsible for visualising the complex three-dimensional structure of the datasets in real-time. It enables users to analyse their datasets intuitively in a fully immersive environment through state of the art visualization techniques including Volume rendering and fast sphere rendering. It is a module developed by HLRS within the COVISE project [17].

Pre-processing module is responsible for preparing the input data together with the corresponding values for the initial and boundary conditions. The required data must meet precise requirements that strongly depend on the considered numerical method.

MD/CFD instances are quite compute and communication intensive and are usually running on MPI and OpenMP. The instantiation of one the simulations are dynamic during the execution of the workflow and the number of instances is depends on the application configuration and output of individual parameters probe.

Post-processing module is responsible for analysing and preparing the output data for visualisation to end-user. It also allows dynamic update of simulation configurations like boundary conditions to run several iterations before desired result is found.
Centralized data storage is mainly used to access persistent input/output files of the applications. The final result of entire parameter sweeps will be aggregated on the centralized storage.

This architecture obviously has to be mapped to the base PaaSage structure. Since the users interact directly with the system through the interfaces of the workflow engine and/or the visualization service, they need to be deployed in a public, or at least shared cloud environment for eScience users and in a private cloud environment for industry users so that the users can access the application from anywhere and at anytime. Taking advantages of the cloud could also ensure the availability and scalability of these modules. The simulation applications might involve different user groups e.g. a university, an institute or a car manufacturer. In order to realize multi-tenancy for serving those multiple user groups (tenants), separate software instances have to be set up. In addition, real-time requirements will necessitate low response time of the according services.

Due to the performance issues mentioned before, the MD/CFD simulation modules have to be deployed to HPC or private cloud that provides compatible performance. The simulation modules will be instantiated at run time and the application code together with input data are to be staged in to allocated compute resources. It has to be noted that, as shown in Figure 6-4, the simulation module itself is a sub-workflow and contains different computation steps that can be roughly categorized into three groups: pre-processing, simulation and post-processing. These steps can be similarly treated as individual logical blocks or modules with individual scaling behaviour. Our long-term goal is to execute the simulation module also across HPC and cloud, for example the embarrassingly parallel workload in cloud and MPI, OpenMP workloads on HPC, but in PaaSage, the simulation applications will be handled as a single module and will be deployed on our HPC.

The pre- and post-processing modules could be deployed to HPC or cloud depends on case, i.e. the requirements of concrete tools/algorithms on capability of the resources. Different instances of pre- or post-processing with different configurations are required for different number of simulations. They should also scale out/in together with the simulation modules to ensure the performance.

Regarding the data storage service, strong consistent cloud storage is required due to the parallel read/write and there is large-volume data transfer (up to several GBs depending on problem size) between the cloud storage and other modules. Depending on the specific case, the results may be shared publicly, in which case the data storage service may be hosted in a public cloud; however, industrial use cases will insist on private deployment and maximum security.

Usage scenario for eScience and Automotive Industry

As mentioned before, to perform a parameter sweep of Molecular Dynamic (MD) or a side mirror development by using CFD:

- Users configure the workflow (number of MD/CFD instances resp. number of approved mirror designs, parameters for each instance, data sources, etc.) and start it by using the workflow engine.
- The tasks within the workflow (simulation instances) are dispatched on various resources through the Internet.
- Input data is pre-processed and is staged in from centralized storage to instances.
- Simulation instances are executed simultaneously.
- The intermediate output data of each simulation will be analysed by post-processing; if certain boundary conditions are fulfilled/violated (e.g. interesting behaviour of molecules is identified resp. undersized field of view or too strong pollution), the post-processing module will inform the workflow engine to interrupt this simulation or other running simulations and restart the workflow with new configurations (e.g. fine granular simulation around the identified point in the parameter space resp. other designs allowing a larger field of view or reduced pollution).
- Final output data is written back to the centralized storage and individual results are combined to form the final results.

- The final results are visualized – ideally in real-time and to multiple users (collaborators) all over the world.
- If a redesign of the simulated model is required, a new workflow with adapted configuration parameters will have to be executed. The users are also able to change the configurations or boundary conditions at runtime i.e. stop the long running simulation and restart it with new configurations automatically to save time and money.

6.5 Walkthrough PaaSage Workflow

Step 0: Offline analysis of codes

As shown in Figure 2-1, the applications will be pre-analyzed by using the PaaSage speculative profiler and stochastic reasoned. The offline analysis will actually take place on basis of a modular application description – similar to the UML diagram. The key point of the analysis consists in checking the dependencies between the modules and in particular between the (non-) functional properties provided per task, respectively on overarching application level. The analysis will use the metadata monitoring profiles to support decomposition of the properties. In Figure 6-4 we have already shown the main modules and their relationship, the specific non-functional and functional properties of each main module are listed as below:

- Workflow Engine
  - Public cloud for eScience users / private cloud for industry users
  - Separate application instance for each user group
  - Relatively few shared users of each instance
  - User number of each user group: 1-10 in eScience case, 10-100 in industry case
  - High availability
  - Small data transfer between workflow engine and other modules, only control messages, events (KB)
- Visualization Web App
  - Public cloud for eScience users / private cloud for industry users
  - Separate application instance for each user group
  - Real time visualization, where response time <= 0.1 second: Threshold limit where users feel that they are directly manipulating objects in the GUI.
  - Relatively few shared users of each instance
  - User number of each user group: 1-10 in eScience case, 10-100 in industry case
  - High availability
  - Medium amount of data (MB) transfer from/to data storage
- Pre-Processing
  - HPC or cloud depends on case
  - Public execution for eScience/ private execution for industry

- MD/CFD
    - HPC because of performance issue
    - Same hardware platforms are required for industry case
    - Public execution for eScience / private execution for industry
    - CPU and data intensive
    - Execution time: from hours to days
    - Large amount (GB to TB) of data transfer from/to data storage
- Post-Processing
    - HPC or cloud depends on case
    - Public execution for eScience / private execution for industry
- Data Storage
    - Public cloud for eScience users / private cloud for industry users
    - Data could be shared between eScience users
    - Data volume: GB to TB, depending on case
    - Location: close to visualization web app due to real time visualization
    - High availability, disaster recovery (replication at more locations)

Step 1: Check the Metadata Database

As shown in Figure 2-1, the metadata is used by PaaSage platform for preparing the deployment and execution of the application. To this end, it actually performs the following tasks: (1) provide decomposition information for step 0, which in turn helps (2) structuring the deployment; and (3) check for availability and match of providers.

Several specific deployment requirements are listed below:

- Existing prototype of Workflow Engine requires .NET, Windows environment; it is a set of WCF services.
- Visualization GUI is platform independent: HTML, AJAX, QT.
- Communication between the Workflow Engine and MD/CFD simulation via SSH, SFTP, SCP.
- MD/CFD requires HPC (MPI, OpenMP).
- Different latencies requirement for different module.
- Different security (public / private) requirements for different user group (eScience, Industry).
- Different sizes of data at different locations.

Step 2: Prepare Deployment

Just prior to deployment, the knowledge of step 0 and step 1 is applied to determine the way of running the application. This implies aspects such as which task runs best where, which ones should be co-located etc., but also, and most importantly, it generates the behavioural instructions for the execution wrapper.

- Appropriate public, private or Hybrid cloud platform should be identified for each Module.
- Select appropriate VM instance size.

- Which modules should be co-located or distributed to different VM.
- Which modules should be scaled, how much and when to ensure the e.g. availability, response time requirements.

Step 3: Deployment and Execution of Scenario

During this step, the code is actively deployed and execution triggered. As depicted in Figure 2-1, during execution, the performance/behaviour and environmental conditions are monitored, analyzed and, if necessary, adaptation steps are taken through the wrapper.

- Possible adaptations
    - MD/CFD is long running jobs within workflow, new instances if fault occurs.
    - (optional: public HPC or cloud will be used if there are no more resources available on private HPC/Cloud)
    - Relocation of in particular data and visualisation services according to user location and/or network experience
- Scale out/in
    - The Workflow engine and visualization should scale out when the response time to user request is bigger than predefined threshold.
    - MD/CFD scales dynamically depending on the configuration in the eScience workflow and output at run time
    - Pre- and post-processing scales together with MD/CFD to ensure performance.
    - Cloud bursting (or rather: HPC bursting), if the number of available resources becomes insufficient
- Where do you get the information from?
    - For real-time visualization: (network) response time
    - Number of available resources in HPC: job queue
- Do you foresee a point when the application may have to be redeployed differently?
    - Not generally, unless the environment fails

Step 4: Monitoring and Completion of Execution, Close-Out Reporting

All monitoring data (cf. step 3) is gathered into dedicated reports that will help improving steps 0-2. In a first approach, as much data as possible is gathered and correlated (such as use case, deployment and actual performance) to build up a knowledge base.

The possible Monitoring data most relevant to eScience application are listed as below:

- Monetary cost, availability, response time
- Application performance (e.g., FLOPS, tasks/sec, MB/sec I/O rates)
- Data transfers rate between cloud storage and HPC
- Execution time / computation speed, T(W,P), where W denotes workload, and P denotes the number of processors/instances
- Speedup (optional)

References

[1] Alexandru Iosup, Simon Ostermann, Nezih Yigitbasi, Radu Prodan, Thomas Fahringer, and Dick Epema. 2011. Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing. IEEE Trans. Parallel Distrib. Syst. 22, 6 (June 2011), 931-945.

[2] Keith R. Jackson, Lavanya Ramakrishnan, Krishna Muriki, Shane Canon, Shreyas Cholia, John Shalf, Harvey J. Wasserman, and Nicholas J. Wright. 2010. Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud. In *Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science*(CLOUDCOM '10). IEEE Computer Society, Washington, DC, USA, 159-168.

[3] Magellan Project, "Magellan Final Report", December 2011

[4] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good. The cost of doing science on the cloud: the montage example. In Proceedings of the 2008 ACM/IEEE conference on Supercomputing, pages 1-12. IEEE Press, 2008.

[5] S. Hazelhurst. Scientific computing using virtual high-performance computing: a case study using the Amazon elastic computing cloud. In Proceedings of the 2008 annual research conference of the South African Institute of Computer Scientists and Information Technologists on IT research in developing countries: riding the wave of technology, pages 94-103. ACM, 2008.

[6] K. Keahey. Cloud Computing for Science. In Proceedings of the 21st International Conference on Scienti_c and Statistical Database Management, page 478. Springer-Verlag, 2009.

[7] K. Keahey, R. Figueiredo, J. Fortes, T. Freeman, and M. Tsugawa. Science clouds: Early experiences in cloud computing for scienti_c applications. Cloud Computing and Applications, 2008, 2008.

[8] M. Palankar, A. Iamnitchi, M. Ripeanu, and S. Gar_nkel. Amazon S3 for science grids: a viable solution? In Proceedings of the 2008 international workshop on Data-aware distributed computing, pages 55-64. ACM, 2008.

[9] Lavanya Ramakrishnan, Keith R. Jackson, Shane Canon, Shreyas Cholia, and John Shalf. 2010. Defining future platform requirements for e-Science clouds. In *Proceedings of the 1st ACM symposium on Cloud computing* (SoCC '10). ACM, New York, NY, USA, 101-106. DOI=10.1145/1807128.1807145 http://doi.acm.org/10.1145/1807128.1807145

[10] Ostermann, S., Iosup, A., Yigitbasi, M.N., Prodan, R., Fahringer, T. & Epema, D.H.J. (2010). Cloud Computing (First International Conference, CloudComp 2009, Munich, Germany, October 19-21, 2009. Revised Selected Papers). In Avresky, D., Diaz, M., Bode, A., Bruno, C. & Dekel, E. (Eds.), *A performance analysis of EC2 cloud computing services for scientific computing*, (*Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 34, pp. 115-131*). Berlin: Springer.

[11] Yan Zhai, Mingliang Liu, Jidong Zhai, Xiaosong Ma, and Wenguang Chen. 2011. Cloud versus in-house cluster: evaluating Amazon cluster compute instances for running MPI applications. In *State of the Practice Reports* (SC '11). ACM, New York, NY, USA, , Article 11 , 10 pages.

[12] Foster, I., Zhao, Y., Raicu, I., and Lu, S. (2008). Cloud Computing and Grid Computing 360-Degree Compared. 2008 Grid Computing Environments Workshop *abs/0901.0*, 1-10. Available at: http://arxiv.org/abs/0901.0131.

[13] Jha, S., Katz, D. S., Luckow, A., Merzky, A. and Stamou, K. (2011) Understanding Scientific Applications for Cloud Environments, in Cloud Computing: Principles and Paradigms (eds R. Buyya, J. Broberg and A. Goscinski), John Wiley & Sons, Inc., Hoboken, NJ, USA. doi: 10.1002/9780470940105.ch13

[14] Windows Workflow Foundation http://msdn.microsoft.com/en-us/vstudio/jj684582.aspx

[15] Windows Communication Foundation http://msdn.microsoft.com/en-us/library/dd456779.aspx

[16] Job Submission Description Language specification http://www.gridforum.org/documents/-GFD.56.pdf

[17] COVISE: http://www.hlrs.de/organization/av/vis/covise/

[18] Call for Large-Scale Projects http://www.gauss-centre.eu/computing-time/call

[19] The Gauss Centre for Supercomputing (GCS) http://www.gauss-centre.eu/

[20] PRACE Research Infrastructure http://www.prace-ri.eu

[21] The European Grid Infrastructure: http://www.egi.eu/

7 Generic Requirements on the PaaSage platform

This section presents requirements on the PaaSage platform that have been generalized from the case study descriptions presented in previous sections. This set of common requirements has been structured in order to drive the design and development process of the PaaSage platform. Those common requirements are maintained in a requirements model from requirements documents can be extracted. In this section we give the current state of this work that will be further consolidated during project year 1 as described in the last section of this report.

This section gives two different views on the requirements:

- A hierarchical decomposition of high level objectives into detailed requirements
- An assignment of requirements to PaaSage components, e.g. profiler or stochastic reasoned.

The requirements were generalised from the case studies in the following way:

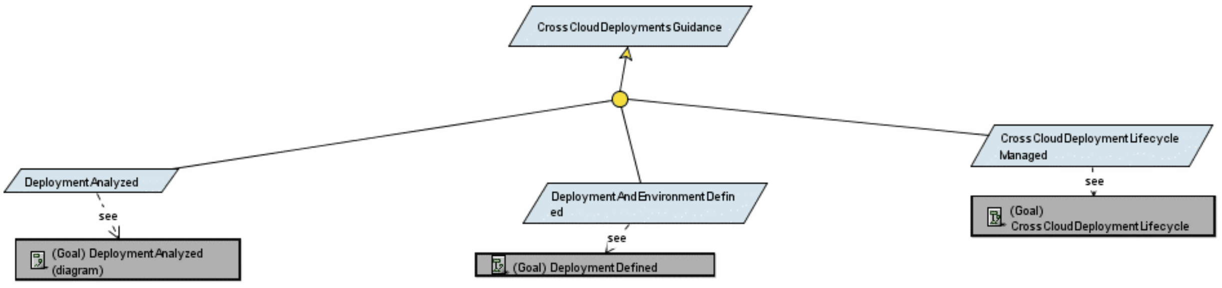
Case study	Requirements
Industrial Sector Case – Flight Scheduling	R-11, R-13, R-15, R-3, R-19, R-22, R-25, R-23, R-1, R-2, R-27, R-5, R-4, R-6, R-33, R-34, R-56, R-57, R-61, R-62, R-60, R-58, R-36, R-39,
Industrial Sector Case – Industrial ERP	R-2
Public sector – electronic portal for citizen-city	R-5, R-17, R-16, R-24, R-18, R-19, R-20, R-21, R-25, R-29, R-2, R-30, R-31, R-57, R-59, R-36, R-40, R-47,
eScience sector – resource intensive simulations	R-1, R-2, R-3, R-4, R-6, R-7, R-8, R-9, R-10, R-12, R-14, R-24, R-28, R-29, R-24, R-2, R-30, R-32, R-52, R-53, R-54, R-55, R-59, R-60, R-35, R-38, R-37, R-40, R-41, R-48, R-42, R-44, R-43, R-45, R-46

Figure 5 Coverage of case studies

7.1 General Requirements

This section aims at inventorying the user requirements by starting from the most strategic goals towards technical requirements needed to achieve them. Each section is structured as follows: there is first an introductory text presenting the section content and commenting the diagram which follows. The only diagrams shown are goal diagrams that show how a higher level objective is decomposed into sub-objectives e.g. in section 7.1.1, and responsibility diagrams that show the requirements that a component is responsible for, e.g. in section 7.2.1

7.1.1 CrossCloudDeploymentsGuidance


Cross Cloud Deployments Guidance
Deployment Analyzed
see
(Goal) Deployment Analyzed (diagram)
Deployment And Environment Defined
see
(Goal) Deployment Defined
Cross Cloud Deployment Lifecycle Managed
see
(Goal) Cross Cloud Deployment Lifecycle

CrossCloudDeploymentsGuidance

The overall objective of the PaaSage platform is to provide guidance in managing cross cloud deployments. To achieve this general objective:

- Deployments must be analysed in order to help the human analyst select the deployment that best meets his requirements: DeploymentAnalyzed (page 58)
- The deployment and the requirements in the target deployment environment must be defined: DeploymentAndEnvironmentDefined (page 59)
- The cross cloud deployment that is selected by the human analyst must be managed by the PaaSage platform throughout the deployment life cycle: CrossCloudDeploymentLifecycleManaged (page 64)

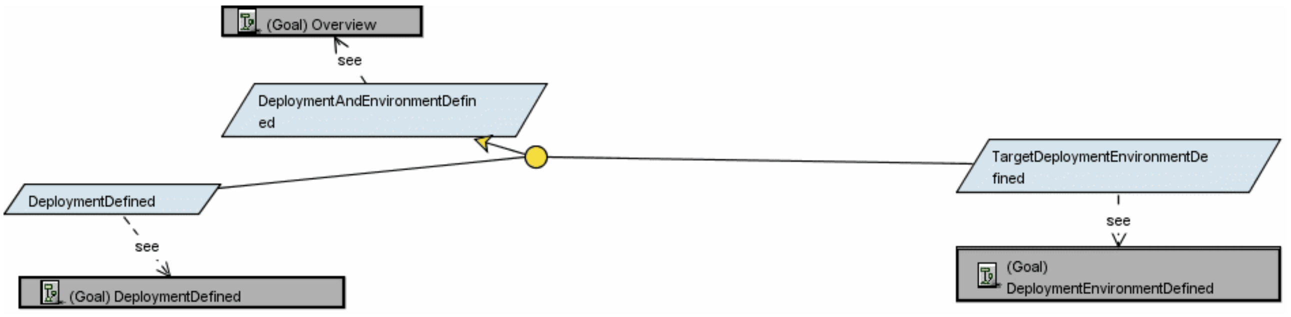
7.1.2 DeploymentAnalyzed

In order to help the human analyst to define his deployment, the PaaSage platform should analyze the application to be deployed and produce a deployment description. To achieve this the following requirements must be satisfied:

Requirement	Agent	Page
R-1 NonFunctionalCriteriaAnalysed The non-functional criteria must be analysed for each module of the software architecture	SpeculativeProfiler	
R-2 ParallelisationCodeAnalysis The platform provides code parallelisation based on an analysis of the code	SpeculativeProfiler	
R-3 ApplicationDependenciesIdentified Analyse the dependencies using the software-architecture information	SpeculativeProfiler	
R-4 ColocationOfVMDefined The modules that need to be co-located or distributed to different VM should be known.	SpeculativeProfiler	
R-5 LegacyApplicationsDeployed Framework for modelling and analysing legacy and cloud applications in order to understand their delivery models and services and find integration solutions	SpeculativeProfiler	
R-6 RequiredCloudTypeKnown The appropriate public, private or hybrid cloud platform should be	SpeculativeProfiler	

identified for each module		
R-7 ProbabilityOfGoodFuturePerformanceKnown provides alternative execution plans based on the probability of good future performance	SpeculativeProfiler	

7.1.3 DeploymentAndEnvironmentDefined

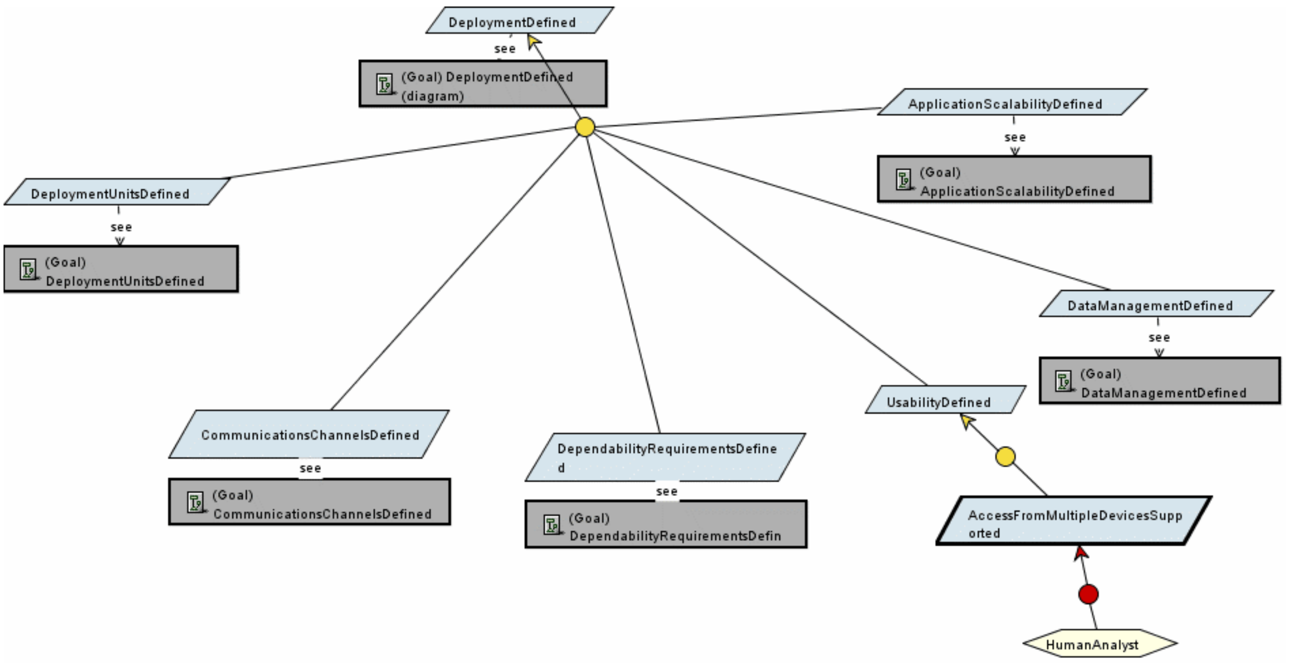

(Goal) Overview
see
DeploymentAndEnvironmentDefined
TargetDeploymentEnvironmentDefined
DeploymentDefined
see
see
(Goal) DeploymentDefined
(Goal) DeploymentEnvironmentDefined

DeploymentAndEnvironmentDefined

The application to be deployed on multiple clouds must be completely defined. This requires:

- Defining the deployment for the application completely: DeploymentDefined (page 59)
- Defining the required cloud environment in which the application must be deployed: TargetDeploymentEnvironmentDefined (page 63)

7.1.4 DeploymentDefined


DeploymentDefined
see
(Goal) DeploymentDefined (diagram)
ApplicationScalabilityDefined
see
(Goal) ApplicationScalabilityDefined
DeploymentUnitsDefined
see
(Goal) DeploymentUnitsDefined
DataManagementDefined
see
(Goal) DataManagementDefined
UsabilityDefined
CommunicationsChannelsDefined
see
(Goal) CommunicationsChannelsDefined
DependabilityRequirementsDefined
see
(Goal) DependabilityRequirementsDefin
AccessFromMultipleDevicesSupported
HumanAnalyst

DeploymentDefined

To describe an application deployment completely the following objectives must be achieved:

- The way in which the application components are aggregated into deployment units must be defined: DeploymentUnitsDefined (page 60)

- The way in which the application components communicate and the required communication channels must be defined: CommunicationsChannelsDefined (page 61)
- The application requirelents in terms of dependability must be defined: DependabilityRequirementsDefined (page 61)
- The manner in which the different deployment units and communications need to scale needs to be defined: ApplicationScalabilityDefined (page 62)
- The requirements on the way that data must be accessed and managed must be described: DataManagementDefined (page 62)

The following requirement must also be taken into account when deploying applications across different clouds:

Requirement	Agent	Page
R-35 AccessFromMultipleDevicesSupported Users have different roles (maybe over time), different knowledge about scheduling insights (e.g. expert schedulers vs. supporting staff) and also different environments where they work. A scheduler can e.g. work in his office using a full-fledged power client or he/she can be in a meeting and needs just read-only access to the data over a mobile device	StochasticReasoner	61

7.1.5 DeploymentUnitsDefined

To define a deployment completely the following requirements must be satisfied:

Requirement	Agent	Page
R-8 DeploymentLocationPreferencesSpecified The preferences for location of deployment units can be specified	HumanAnalyst	
R-9 PreferredDataPlacementLocationDefined Different sizes of data at different locations	HumanAnalyst	
R-10 AproprivateVMInstanceSizeTypeKnown The appropriate VM instance size and type must be selected	HumanAnalyst	
R-11 PhysicalEnvironmentsMappedToPlatform a platform-specific mapping is required for all physical environments	StochasticReasoner	
R-12 RequiredComponentLatenciesKnown The different latencies required for different modules should be known	HumanAnalyst	
R-13 TransientWorflowsSupported This new model supports a kind of a 'transient workflow', which means that everything he/she does is persistent and available on whatever client he/she works on. When a user moves e.g. from the desktop browser to a mobile client, he/she expects to see the same data after login to the same application.	StochasticReasoner	
R-14 MinCostForMaxPerformanceOfWorkflow		

Minimize the leasing cost while maximizing the contribution to reducing the overall workflow execution.	StochasticReasoner	
R-15 AccessFromMultipleDevicesSupported Users have different roles (maybe over time), different knowledge about scheduling insights (e.g. expert schedulers vs. supporting staff) and also different environments where they work. A scheduler can e.g. work in his office using a full-fledged power client or he/she can be in a meeting and needs just read-only access to the data over a mobile device	StochasticReasoner	

7.1.6 CommunicationsChannelsDefined

In order to define the communications channels that are need for communication between the different deployment units, the following requirements must be satisfied:

Requirement	Agent	Page
R-3 InteroperabilityBetweenApplicationDefined It should be possible to increased interoperability between applications, at least for applications of the same application suite	HumanAnalyst	
R-16 CrossCloudCommunicationSupported Deployments (Services) must therefore be able to communicate seamlessly across different cloud-based applications	StochasticReasoner	
R-17 SeamlessMultipleCloudIntegrationSupported It should be possible for example to support a process where a workflow and User Interface is run in a private cloud, but it reuses public data/Open Data-databases, and integrated with locally installed archiving and accounting systems for a municipality.	StochasticReasoner	
R-24 CloudServiceIntegratedWithCustomerApplications It should be possible for existing applications with a large local installed base to integrate these with a cloud offering delivering standardized processes where the process is run in the cloud, but closely integrated with the business applications installed at each individual customer	HumanAnalyst	66

7.1.7 DependabilityRequirementsDefined

In order to describe the dependability that is required for the deployment, the following requirements must be satisfied :

Requirement	Agent	Page
R-18 CrossCloudAccessControlIntegrated Support methods to ensure access control across datasources	StochasticReasoner	
R-19 EndToEndSecurityGuaranteed Security concerns must be covered at all time, moving from a private cloud e.g. into a public cloud (even for parts of the system) must be possible in a secure and reliable way. It must provide	StochasticReasoner	

trustworthy services.		
R-20 CrossCloudDataIntegrityAndAuthenticity The integrity and authenticity of data should be guaranteed end-to end	StochasticReasoner	
R-21 EndToEndDataIntegrityGuaranteed data integrity must be guaranteed	StochasticReasoner	
R-22 ServicesAvailableGlobally Deployed services should be available globally.	StochasticReasoner	
R-25 HighAvailabilityOfServices Access to external interfaces is a vital part for such deployments. Data can be exported and imported using a standard file format and data can be sent to other departments or to partners. High availability is important.	StochasticReasoner	66
R-23 OperationalIntegrityGuaranteed Data must be constantly updated but operational integrity must be maintained.	StochasticReasoner	

7.1.8 ApplicationScalabilityDefined

The way in which the different deployment units and communication channels must scale across the different clouds must be defined:

Requirement	**Agent**	**Page**
R-24 ModuleScalabilityDefined It should be specified which modules need to scale, how much and when to ensure the e.g. availability and response time requirements	HumanAnalyst	
R-25 OtherScalabilityDefined Scalability other than elasticity must also be defined, e.g. defining how much memory could be allocated to an application.	HumanAnalyst	
R-2 ElasticityDefined Elasticity and scalability across datacenters, and across business processes over the year should be specified	HumanAnalyst	
R-26 ApplicationLoadDefined The expected application load for an application should be defined	HumanAnalyst	
R-27 SmallAndLargeCustomersServed Deployed systems should be available every day by many customers around the globe, ranging from small to large companies and using different business models.	StochasticReasoner	

7.1.9 DataManagementDefined

The deployment definition must include a description of all data that needs to be managed by the application deployed in the cloud:

Requirement	Agent	Page
R-1 DatabaseScalabilityDefined The sizing of the servers must be done up front and elasticity of the database servers must be anticipated (e.g. transform an Oracle single node database server into a cluster (RAC) database server)	HumanAnalyst	
R-28 DataVolumeSpecified Expected data volumes must be specified	HumanAnalyst	
R-29 CrossCloudDataFlowModelled It should be possible to model processes and dataflow across cloud and local solutions	HumanAnalyst	

7.1.10 TargetDeploymentEnvironmentDefined

The target cloud environment in which the application must be deployed should be defined. The description of the target environment should include the following:

Requirement	Agent	Page
R-5 TestEnvironmentStrictlySeperatedFromOperationalApplication Test systems are as close as possible to the real application, but still strictly separated; just another instance in the cloud	StochasticReasoner	
R-4 TestEnvironmentsEasilySetUp Easy setup of different test environments for different test scenarios (e.g. RfC tests, exploration of new business scenarios, integration tests etc.)	HumanAnalyst	
R-30 HybridCloudDeploymentSupported hybrid cloud models should be supported with some services in private clouds and some services in public clouds	StochasticReasoner	
R-31 GradualMigrationToCloudSupported Moving to the cloud will contain trial and error experiences where applications are gradually shifted from locally installed software to gradually more cloud based models	StochasticReasoner	
R-6 MigrationFromTestEnvironmentSupported Configuring and migrating an application to a new environment is a long and error prone process. These migration steps needs to be executed in a test environment beforehand	StochasticReasoner	
R-32 RequiredCloudForEachModuleKnown The appropriate public, private or hybrid platform should be identified for each module.	HumanAnalyst	
R-33 ServerDeployedInCustomerPrivateCloud The RDBMS and the application server(s) can be deployed and operated in our data centre or in a customer's data centre	StochasticReasoner	
R-34 ServerDeployedInPrivateCloud Applications should be deployable in a private cloud		

	StochasticReasoner	

7.1.11 CrossCloudDeploymentLifecycleManaged

Once the deployment of an application is completely defined the PaaSage platform should help to manage the whole deployment lifecycle of the application. This requires helping the human analyst to select a good deployment (GoodDeploymentsProposed page 65) and satisfying the following requirements:

Requirement	Agent	Page
R-35 DeploymentSelected The human analyst must be able to select a deployment from several deployment scenarios and to easily understand the tradeoffs between the different deployments.	HumanAnalyst	
R-36 FullPortabilityMaintained Full portability of the cross cloud deployments must be guaranteed	CrossCloudDeploymentManager	
R-37 MinCloudAdministrativeOverhead easier access to resources that minimizes the administrative overhead	StochasticReasoner	
R-38 AvailabilityOfComponentsMonitored Availability of deployed components must be monitored.	CrossCloudDeploymentManager	
R-39 CloudNetworkOptimisationsSupported performance depends on the network connection into the cloud. The new architecture should use cloud specific network optimizations	StochasticReasoner	
R-40 ResponseTimesMonitored The response time of deployed components must be monitored	CrossCloudDeploymentManager	
R-41 RelocationBasedOnUserExperience relocation of deployed services and data based on user experience	Adapter	
R-42 RelocationbasedonNetworkExperience Relocation of servoces and data based on network experience	Adapter	
R-43 AdaptationGuidedByPolicies Adaptation, which is an automatic process, should be guided by policies; for instance should we halt and migrate some VMs to another cloud provider, or just continue running sub-optimally?		
R-44 CloudBurstingSupported Cloud bursting should be possible	CrossCloudDeploymentManager	
R-45 DeploymentsReconfigured During execution, there is real-time checking whether the	Adapter	

performance is as you expect (via updates to the MD-DB). Several options when SLA is violated, prioritise alternative resources. If performance drops below acceptable levels in the SLA – Halt maybe check point and reconfigure		
R-46 InstancesRestarted VM instances should be restarted when faults occur	CrossCloudDeploymentManager	
R-47 IntegratedCrossCloudDeploymentManagement Framework for «SOA/Cloud» management should keep control on dependencies	CrossCloudDeploymentManager	
R-48 DeploymentReportFinalised After execution has been completed a scenario close out report on overall performance will need to be lodged with the MD-DB	CrossCloudDeploymentManager	

7.1.12 GoodDeploymentsProposed

In order to help the human analyst to select a deployment that best meets his needs the PaaSage platform should propose some possible deployments based on the deployment definition. To achieve this objective the following requirements should be met:

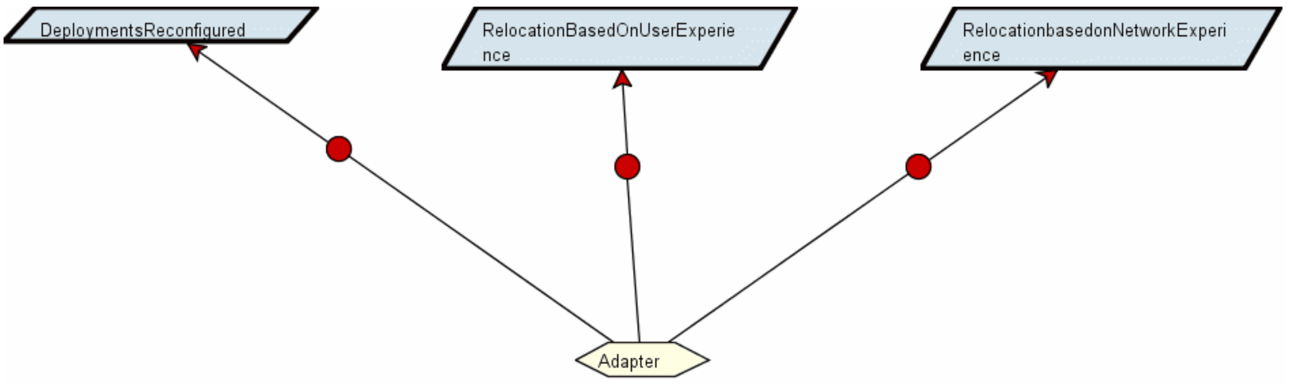
Requirement	**Agent**	**Page**
R-49 CloudProvidersKnown A list of available cloud providers should be known be managed. Types of cloud providers should be captured, e.g. Enterprise Software Bus as a service.	StochasticReasoner	
R-50 NearOptimalDeploymentCalculated The deployments that are calculated do not have to be optimal, but should be near optimal.	StochasticReasoner	
R-51 TargetDeploymentEnvMappedToCloudProviders The required target environment should be mapped to the target cloud providers. This mapping needs to be managed across the deployment lifecycle.	StochasticReasoner	
R-52 ReputationOfCloudProvidersTakenIntoAccount The reputation of available cloud providers should be managed: it should be based on past performance	StochasticReasoner	
R-53 ImprovedDistributionOfLoad The application load should be distributed accross the cloud resources.	StochasticReasoner	
R-54 CostTimeTradeoffsTakenIntoAccount		

Different types of tradeoffs should be captured: - Capture Cost/time tradeoffs, use of private/public clouds (private may be preferred if available – e.g. security may demand that handling certain data cannot be removed from the Private cloud).	StochasticReasoner	
R-55 PriorityOfRequestTakenIntoAccount Take into account the urgency and priority of the request.	StochasticReasoner	
R-56 CloudEnabledDataManagement The database technology used in a cloud environment needs to be a different one . Topics like the CAP theorem, ACID vs. BEST, the shared-nothing approach etc. needs to be addressed in such application architecture, designed for the cloud.	StochasticReasoner	
R-57 ExternalDataAccessible It should be possible to easily access other IT-systems within a company and outside of the company	StochasticReasoner	
R-58 TimeZonesSupported Time zones should be taken into account when deploying an application in multiple clouds, especially when the application must be accessible globally from anywhere in the world.	StochasticReasoner	
R-59 CloudServiceIntegratedWithCustomerApplications It should be possible for existing applications with a large local installed base to integrate these with a cloud offering delivering standardized processes where the process is run in the cloud, but closely integrated with the business applications installed at each individual customer	HumanAnalyst	
R-2 ElasticityDefined Elasticity and scalability across datacenters, and across business processes over the year should be specified	HumanAnalyst	
R-60 HighAvailabilityOfServices Access to external interfaces is a vital part for such deployments. Data can be exported and imported using a standard file format and data can be sent to other departments or to partners. High availability is important.	StochasticReasoner	
R-61 PayPerUseAccountingModel A pay-per-use model must be used. Several models should be investigated such as pay per use, pay as you save, pay one time access fee, or a mix of other models. The issue of aggregation of payment model must be addressed when several providers are involved.	StochasticReasoner	
R-62 CostFunctionKnown The cost function of a cloud deployment should be known to the customers so that they can estimate costs based on different load scenario.	StochasticReasoner	

7.2 Component Responsibilities

This section of the SRS lists again all the requirements stated in the first part of the document. They are classified this time according to the agents who are responsible for them. The system agents are first listed, then the environment agents, and finally, if needed, the undefined category agents. Each requirement stated is associated with a page number referring to the first part of this document where this requirement appears for the first time. Additionally, this section can also contain conceptual descriptions regarding the application domain or the system.

7.2.1 Adapter

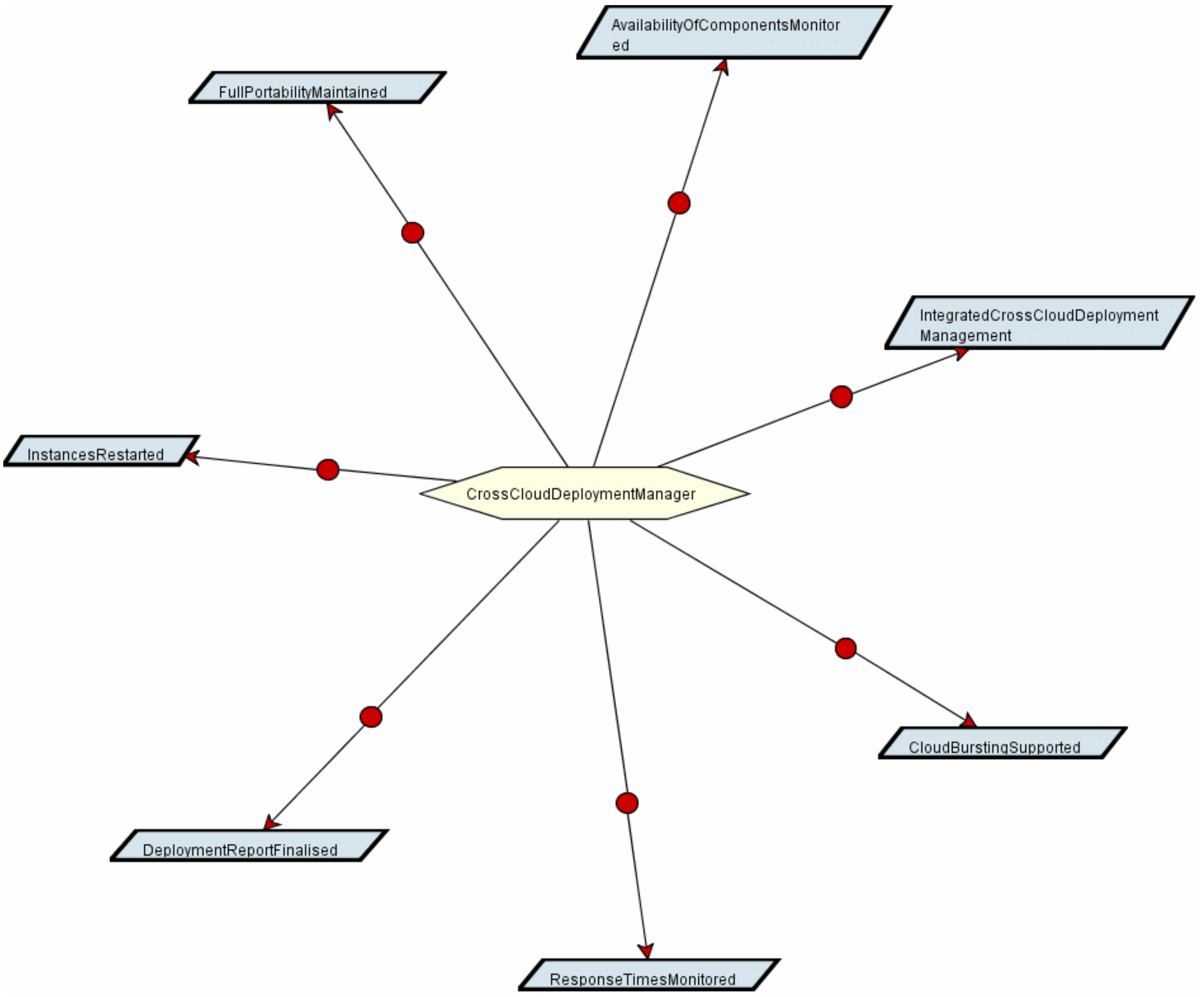


DeploymentsReconfigured
RelocationBasedOnUserExperience
RelocationbasedonNetworkExperience
Adapter

List of Responsibilities :

Requirement	Page
R-61 RelocationBasedOnUserExperience relocation of deployed services and data based on user experience	64
R-65 DeploymentsReconfigured During execution, there is real-time checking whether the performance is as you expect (via updates to the MD-DB). Several options when SLA is violated, prioritise alternative resources. If performance drops below acceptable levels in the SLA – Halt maybe check point and reconfigure	64
R-62 RelocationbasedonNetworkExperience Relocation of servoces and data based on network experience	64

7.2.2 CrossCloudDeploymentManager

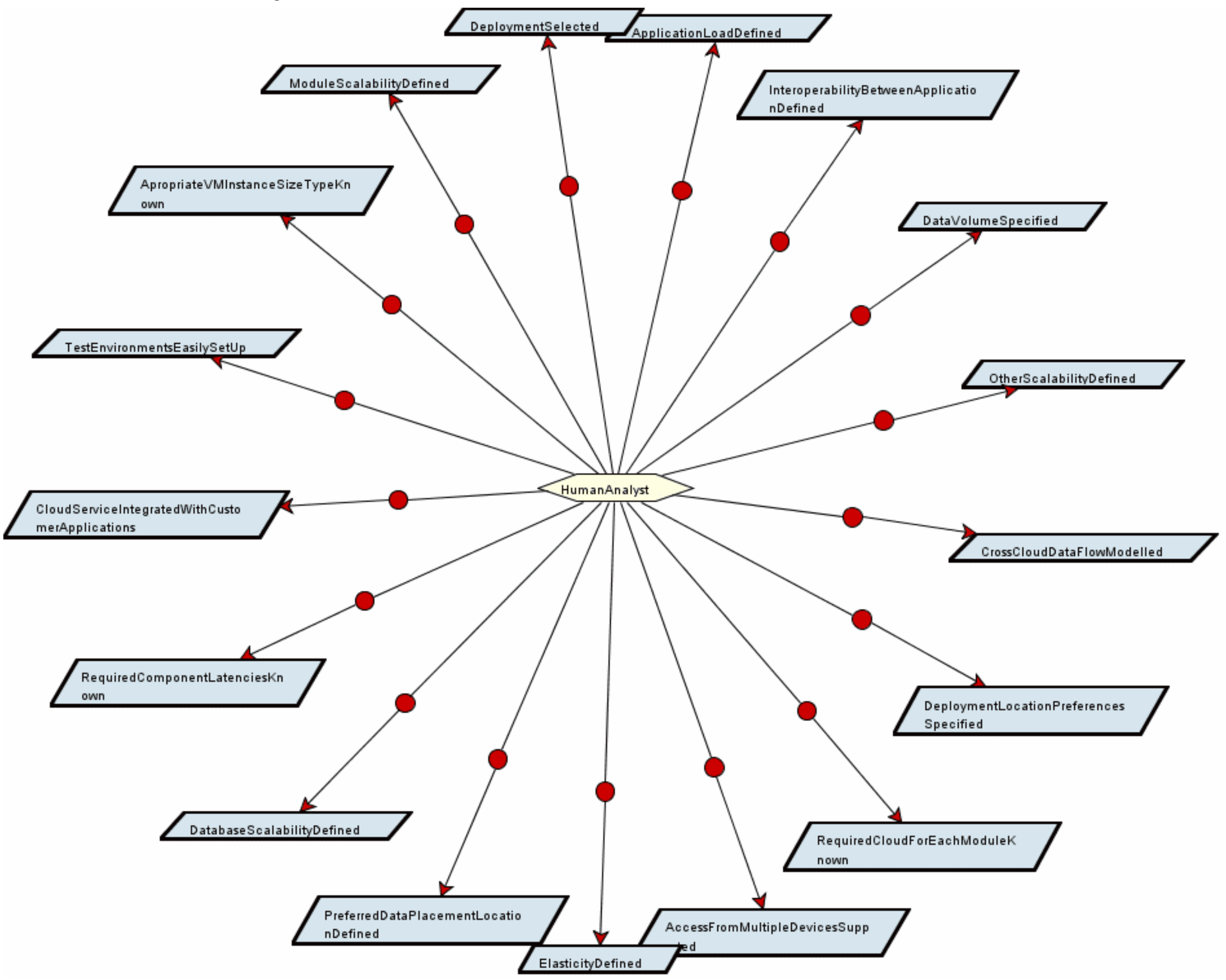

AvailabilityOfComponentsMonitored
FullPortabilityMaintained
IntegratedCrossCloudDeploymentManagement
InstancesRestarted
CrossCloudDeploymentManager
CloudBurstingSupported
DeploymentReportFinalised
ResponseTimesMonitored

List of Responsibilities :

Requirement	Page
R-67 IntegratedCrossCloudDeploymentManagement Framework for «SOA/Cloud» management should keep control on dependencies	65
R-58 AvailabilityOfComponentsMonitored Availability of deployed components must be monitored.	64
R-60 ResponseTimesMonitored The response time of deployed components must be monitored	64
R-56 FullPortabilityMaintained Full portability of the cross cloud deployments must be guaranteed	64
R-64 CloudBurstingSupported Cloud bursting should be possible	64
R-68 DeploymentReportFinalised After execution has been completed a scenario to close out report on overall performance will need to be lodged with the MD-DB	65
R-66 InstancesRestarted	

VM instances should be restarted when faults occur	65

7.2.3 HumanAnalyst

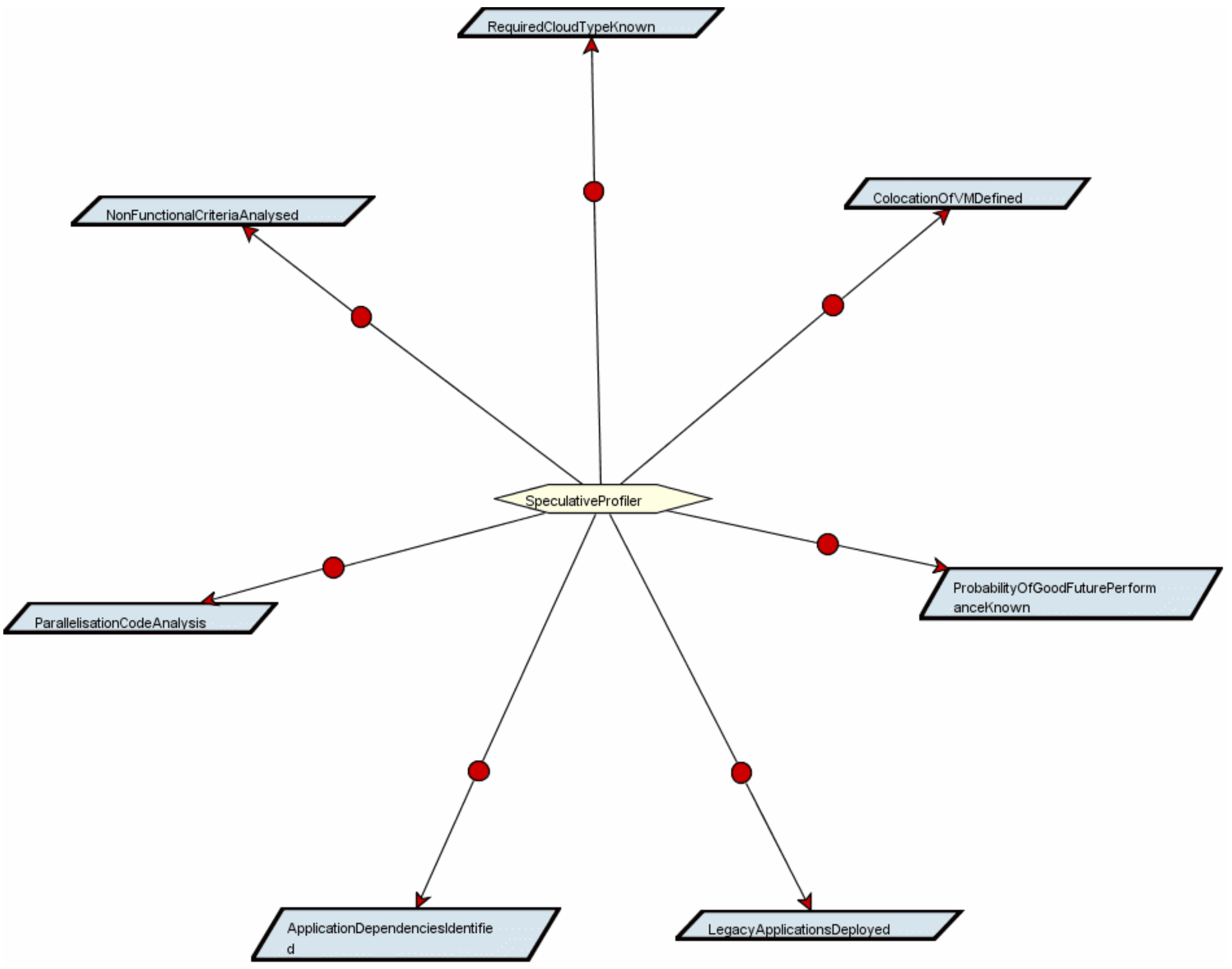

DeploymentSelected
ApplicationLoadDefined
ModuleScalabilityDefined
InteroperabilityBetweenApplicatio nDefined
ApropriateVMInstanceSizeTypeKn own
DataVolumeSpecified
TestEnvironmentsEasilySetUp
OtherScalabilityDefined
HumanAnalyst
CloudServiceIntegratedWithCusto merApplications
CrossCloudDataFlowModelled
RequiredComponentLatenciesKn own
DeploymentLocationPreferences Specified
DatabaseScalabilityDefined
RequiredCloudForEachModuleK nown
PreferredDataPlacementLocatio nDefined
AccessFromMultipleDevicesSupp ed
ElasticityDefined

List of Responsibilities :

Requirement	**Page**
R-46 ApplicationLoadDefined The expected application load for an application should be defined	62
R-3 InteroperabilityBetweenApplicationDefined It should be possible to increased interoperability between applications, at least for applications of the same application suite	
R-49 CrossCloudDataFlowModelled It should be possible to model processes and dataflow across cloud and local solutions	63
R-24 CloudServiceIntegratedWithCustomerApplications It should be possible for existing applications with a large local installed base to integrate these with a cloud offering delivering standardized processes where the process is run in the cloud, but closely integrated with the business applications installed at each individual customer	66
R-35 AccessFromMultipleDevicesSupported	

Users have different roles (maybe over time), different knowledge about scheduling insights (e.g. expert schedulers vs. supporting staff) and also different environments where they work. A scheduler can e.g. work in his office using a full-fledged power client or he/she can be in a meeting and needs just read-only access to the data over a mobile device	61
R-28 DeploymentLocationPreferencesSpecified The preferences for location of deployment units can be specified	60
R-45 OtherScalabilityDefined Scalability other than elasticity must also be defined, e.g. defining how much memory could be allocated to an application.	62
R-52 RequiredCloudForEachModuleKnown The appropriate public, private or hybrid platform should be identified for each module.	63
R-32 RequiredComponentLatenciesKnown The different latencies required for different modules should be known	60
R-48 DataVolumeSpecified Expected data volumes must be specified	63
R-4 TestEnvironmentsEasilySetUp Easy setup of different test environments for different test scenarios (e.g. RfC tests, exploration of new business scenarios, integration tests etc.)	
R-29 PreferredDataPlacementLocationDefined Different sizes of data at different locations	60
R-44 ModuleScalabilityDefined It should be specified which modules need to scale, how much and when to ensure the e.g. availability and response time requirements	62
R-30 AproprLateVMInstanceSizeTypeKnown The appropriate VM instance size and type must be selected	60
R-2 ElasticityDefined Elasticity and scalability across datacenters, and across business processes over the year should be specified	
R-55 DeploymentSelected The human analyst must be able to select a deployment from several deployment scenarios and to easily understand the tradeoffs between the different deployments.	64
R-1 DatabaseScalabilityDefined The sizing of the servers must be done up front and elasticity of the database servers must be anticipated (e.g. transform an Oracle single node database server into a cluster (RAC) database server)	

7.2.4 SpeculativeProfiler


RequiredCloudTypeKnown
NonFunctionalCriteriaAnalysed
ColocationOfVMDefined
SpeculativeProfiler
ParallelisationCodeAnalysis
ProbabilityOfGoodFuturePerformanceKnown
ApplicationDependenciesIdentified
LegacyApplicationsDeployed

List of Responsibilities :

Requirement - Expectation	Page
R-12 RequiredCloudTypeKnown The appropriate public, private or hybrid cloud platform should be identified for each module	58
R-10 ColocationOfVMDefined The modules that need to be co-located or distributed to different VM should be known.	58
R-9 ApplicationDependenciesIdentified The dependencies should be analyzed using the software-architecture information	58
R-8 ParallelisationCodeAnalysis Code parallelisation should be provided based on an analysis of the code	58
R-13 ProbabilityOfGoodFuturePerformanceKnown Alternative execution plans should be provided based on the probability of good future performance	59
R-7 NonFunctionalCriteriaAnalysed The non-functional criteria must be analyzed for each module of the software architecture	58
R-11 LegacyApplicationsDeployed	

A framework for modeling and analyzing legacy and Cloud applications in order to understand their delivery models and services and find integration solutions should be defined.	58

7.2.5 StochasticReasoner

StochasticReasoner has more than 30 responsibilities, which could not be represented on a single responsibility diagram and keep it readable.

List of Responsibilities :

Requirement	Page
R-36 CrossCloudCommunicationSupported Deployments (Services) must therefore be able to communicate seamlessly across different cloud-based applications	61
R-37 SeamlessMultipleCloudIntegrationSupported It should be possible for example to support a process where a workflow and User Interface is run in a private cloud, but it reuses public data/Open Data-databases, and integrated with locally installed archiving and accounting systems for a municipality.	61
R-19 CostTimeTradeoffsTakenIntoAccount Different types of tradeoffs should be captured: - Capture Cost/time tradeoffs, use of private/public clouds (private may be preferred if available – e.g. security may demand that handling certain data cannot be removed from the Private cloud).	65
R-40 CrossCloudDataIntegrityAndAuthenticity The integrity and authenticity of data should be guaranteed end-to end	62
R-34 MinCostForMaxPerformanceOfWorkflow Minimize the leasing cost while maximizing the contribution to reducing the overall workflow execution.	60
R-20 PriorityOfRequestTakenIntoAccount The urgency and priority of the request should be taken into account	66
R-51 GradualMigrationToCloudSupported The platform should provide support for moving to the cloud because it will contain trial and error experiences where applications are gradually shifted from locally installed software to gradually more cloud based models	63
R-35 AccessFromMultipleDevicesSupported Users have different roles (maybe over time), different knowledge about scheduling insights (e.g. expert schedulers vs. supporting staff) and also different environments where they work. A scheduler can e.g. work in his office using a full-fledged power client or he/she can be in a meeting and needs just read-only access to the data over a mobile device	61
R-39 CloudNetworkOptimisationsSupported Performance depends on the network connection into the cloud. The new architecture should use cloud specific network optimizations	64
R-42 ServicesAvailableGlobally Deployed services should be available globally.	62

R-18 ImprovedDistributionOfLoad The application load should be distributed across the cloud resources.	65
R-22 ExternalDataAccessible It should be possible to easily access other IT-systems within a company and outside of the company	66
R-21 CloudEnabledDataManagement The database technology used in a cloud environment needs to be a different one . Topics like the CAP theorem, ACID vs. BEST, the shared-nothing approach etc. needs to be addressed in such application architecture, designed for the cloud.	66
R-14 CloudProvidersKnown A list of available cloud providers should be known be managed. Types of cloud proiders should be captured, e.g. Enterprise Software Bus as a service.	65
R-47 SmallAndLargeCustomersServed Deployed systems should be available every day by many customers around the globe, ranging from small to large companies and using different business models.	62
R-27 CostFunctionKnown The cost function of a cloud deployment should be known to the customers so that they can estimate costs based on different load scenario.	66
R-5 TestEnvironmentStrictlySeperatedFromOperationalApplication Test systems are as close as possible to the real application, but still strictly separated; just another instance in the cloud	
R-16 TargetDeploymentEnvMappedToCloudProviders The required target environment should be mapped to the target cloud providers. This mapping needs to be managed across the deployment lifecycle.	65
R-25 HighAvailabilityOfServices Access to external interfaces is a vital part for such deployments. Data can be exported and imported using a standard file format and data can be sent to other departments or to partners. High availability is important.	66
R-57 MinCloudAdministrativeOverhead easier access to resources that minimizes the administrative overhead	64
R-6 MigrationFromTestEnvironmentSupported Configuring and migrating an application to a new environment is a long and error prone process. These migration steps needs to be executed in a test environment beforehand	
R-17 ReputationOfCloudProvidersTakenIntoAccount The reputation of available cloud providers should be managed: it should be based on past performance	65
R-38 CrossCloudAccessControlIntegrated Support methods to ensure access control across datasources	61
R-15 NearOptimalDeploymentCalculated The deployments that are calculated do not have to be optimal, but should be near optimal.	65
R-50 HybridCloudDeploymentSupported	

hybrid cloud models should be supported with some services in private clouds and some services in public clouds	63
R-43 OperationalIntegrityGuaranteed Data must be constantly updated but operational integrity must be maintained.	62
R-26 PayPerUseAccountingModel A pay-per-use model must be used.	66
R-41 EndToEndDataIntegrityGuaranteed data integrity must be guaranteed	62
R-33 TransientWorflowsSupported This new model supports a kind of a 'transient workflow', which means that everything he/she does is persistent and available on whatever client he/she works on. When a user moves e.g. from the desktop browser to a mobile client, he/she expects to see the same data after login to the same application.	60
R-53 ServerDeployedInCustomerPrivateCloud The RDBMS and the application server(s) can be deployed and operated in our data centre or in a customer's data centre	63
R-23 TimeZonesSupported	66
R-39 EndToEndSecurityGuaranteed Security concerns must be covered at all time, moving from a private cloud e.g. into a public cloud (even for parts of the system) must be possible in a secure and reliable way. It must provide trustworthy services.	61
R-54 ServerDeployedInPrivateCloud Applications should be deployable in a private cloud	63
R-31 PhysicalEnvironmentsMappedToPlatform a platform-specific mapping is required for all physical environments	60

These requirements models has been derived from the cases study descriptions (see Figure 5). After the initla requirements analysis these requirements will need to be refined further, and the coverage of the cases studies will need to be validated to ensure that the requirements are complete with respect to the case studies.

8 Outlook

This initial requirements deliverable focused on the description of requirements of the cloud deployment case studies in different applications domains. As it is also the result of the first months of cooperation between the project partners with different expertise and backgrounds (domain experts, technical expert, scientists etc.) it was important to achieve a common vision among them. The approach taken was to understand what a cloud deployment meant for the different case studies. The case studies started from a set of concrete scenarios in the different application domains available within the project and defined how they could be deployed using the high level PaaSage framework and architecture available at this point. This allowed us to identify a number of requirements in each domain. From those case study specific requirements a first consolidation and generalisation into a set of common requirements was done.

Based on this the next steps of the requirements and specification process will be to:

- First present the documented scenarios and their mapping on the PaaSage workflow to the rest of the PaaSage partners. This work will more specifically managed in coordination with the architecture WP.
- Based on the collected feedback, both the architecture and the requirements will be refined in each domain. In parallel, the common set of requirements already produced will be further elaborated and documented. An important outcome that will be achieved at this step is the further elaboration of the CloudML deployment language.
- To support the development process in specific WP tasks, the requirements will be assigned to specific components and will yield the specification of those components. This will cover both functional and non-functional requirements.
- To support the validation and test process as well as the demonstrators and exploitation plan, the requirements will be formulated in measurable terms, typically by associating a satisfaction (or fit) criteria to each requirement. This will result in a validation plan that will be used later on in WP6 to check to what extend the resulting platforms meets its requirements.

As the initial set of requirements presented in this document will evolve, WP6 will keep track of new requirements, more precise descriptions, possible change in priorities, etc. This process will be managed internally and will be reflected in the final requirements deliverable D.6.1.2 due at month 24 (project mid-term). At that time requirements are expected to have stabilized.