Grant Agreement N° 215483

Title:	*Detailed requirements analysis for a Distributed Services Laboratory Network*
Authors:	*CNR, Tilburg, UniDue, TUW, UPM*
Editor:	*Michele Mancioppi (Tilburg)*
Reviewers:	*Ita Richardson (Lero)*
	Manuel Carro Liñares (UPM)
Identifier:	*Deliverable CD-IA-1.2.2*
Type:	*Contractual Deliverable*
Version:	*1*
Date:	*14 March 2009*
Status:	*Final*
Class:	*External*

Management Summary

The *Pan-European Distributed Service Laboratory* (EDSL) is a joint effort from the S-Cube partners to realize a shared, versatile and flexible infrastructure to conduct experiments with service-related technologies. The EDSL will provide the appropriate infrastructure to enable the correct setup of shared experiments by allowing an efficient dissemination of knowledge and experimental results. Furthermore, it will promote collaboration, reuse and eventual integration of research results and allow validation of research findings. The realization of the EDSL will require the small-scale, lightweight integration of technologies and software that can be used in conjunction with computing facilities provided by S-Cube partners.

This deliverable presents the S-Cube vision for the EDSL in terms of desired characteristics and functions offered. The vision is exemplified by a comprehensive use case based on the S-Cube "Supply Chain Scenario". Starting from the vision and the use case, the deliverable outlines the requirements that result from the analysis effort conducted by the S-Cube participants in months 6-12 of the network. The requirements here reported will serve as a basis to realize the first prototype of the EDSL by month 23. Finally, the deliverable proposes a timeline that relates the foreseen EDSL realization process with the upcoming deliverables of the IA-1.2 workpackage.

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

Members of the S-Cube consortium:

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

Published S-Cube documents

These documents are all available from the project website located at http://www.s-cube-network.eu/

- CD-IA-1.1.1 Comprehensive Overview of the State of the Art on Service-based Systems
- CD-SoE-1.2.2 Collaboration Plan for Joint Activities with ICT SSAI&E Projects
- PO-IA-1.2.1 Definition of the Collaboration Infrastructure and Identification of Appropriate Tools for Exchange of Knowledge
- PO-IA-2.1.1 Definition and Set-up of Mobility Program Policies
- PO-IA-2.2.1 Identification of Potential Industrial Collaborators
- PO-JRA-1.1.1 State of the art report on software engineering design knowledge and Survey of HCI and contextual knowledge
- PO-JRA-1.2.1 State-of-the-Art report on principles, techniques and methodologies for monitoring and adaptation
- PO-JRA-1.3.1 Survey of quality related aspects relevant for SBAs
- PO-JRA-2.1.1 State-of-the-art survey on business process modelling and Management
- PO-JRA-2.2.1 Overview of the state of the art in composition and coordination of services
- PO-JRA-2.3.1 Use Case Description and State-of-the-Art
- PO-SoE-1.1.1 Analysis of Current Programmes

The S-Cube Deliverable Series

Vision and Objectives of S-Cube

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

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

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

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

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

Contents

1 Introduction

S-Cube aims at re-aligning, re-shaping and integrating the research agendas of different key European players from different research areas to empower Europe to lead the software-services revolution. This integration will be carried out along three main axes: integration and synthesis of diversified knowledge in the areas of Business Process Models, Service Engineering, Service Oriented Computing and Grid Computing, integration of existing research infrastructures from S-Cube beneficiaries and alignment and rationalisation of European education and training programs.

One of the objectives of S-Cube is to share existing research infrastructures of the partners into a common, shared facility, thus creating new opportunities for research, learning and experimentation. S-Cube beneficiaries will share resources (e.g., computing facilities, data, ICT infrastructures, etc.) across geographic and institutional boundaries. The shared resources will be linked to form the *Pan-European Distributed Service Laboratory* (EDSL), which will enable the trial, evaluation, and integration of novel service-related concepts, technologies and system solutions. The EDSL will facilitate close collaboration and eventual integration among researchers from the Service Oriented Computing, Software Engineering, Grid Computing and Business Process Management communities. This will be achieved by allowing the linking of diverse partner infrastructures to provide a common environment for the demonstrations of research results. Through the use of the EDSL, the S-Cube beneficiaries will have the ability to access various items such as tools, demonstrators, services and virtualisation environments. These facilities will be used to test and experiment with S-Cube results, such as service engineering methodologies, cross-layer adaptation mechanisms, use of the service oriented infrastructure, service architectures, service composition and coordination mechanisms and facilities for the management of service-enabled business processes.

In summary, the EDSL will provide the appropriate infrastructure to enable the correct setup of shared experiments by allowing an efficient dissemination of knowledge and experimental results. Furthermore, it will promote collaboration, reuse and eventual integration of research results and allow validation of research findings.

This current deliverable focuses on presenting the requirements that outline the functions the EDSL will provide to its users and how those users will interact with the EDSL. In addition to the high-level requirements, this document presents a timeline fro the future development of the EDSL, taking the deliverables for the project into account.

Which kind of Requirements?

In this deliverable we will describe the functions that the EDSL provides in terms of user requirements (or simply requirements in the remainder) that summarise the outcome of the ongoing discussion that has involved the S-Cube participants over the past 6 months. Note the term requirement is used in the software community and industry inconsistently, therefore the definition of requirement we adopt is the following:

> User requirements are statements, in a natural language (possibly plus diagrams), of what services the system is expected to provide and the constraints under which it must operate [4].

Which kind of Experiments?

This deliverable uses the term “experiment” as the execution of a software demonstrator from which EDSL users want to extract results that will be included in S-Cube related publications. In particular, the EDSL will support experimental processes aiming at verifying research hypothesis.

Therefore, our use of the term experiment has a different connotation than in the E-Science community. Furthermore, the experiments we take into account are not meant to test a software applications or

systems, e.g. for the purpose of Quality Assurance.

Structure of the Deliverable

The deliverable is structured as follows: Section 2 formalises the vision for the EDSL, outlining its role and intended function in S-Cube and a high-level view of the functions it provides. Section 3 presents the EDSL's requirements that have been elicited by the S-Cube partners, and connects them with the vision presented in Section 2. Section 4 presents an overview of the future work in the Work-Package IA-1.2 as a timeline for the EDSL. Finally, Section 5 concludes the deliverable by presenting our conclusions.

2 The Vision for the Pan-European Distributed Service Laboratory

The EDSL is the effort of the S-Cube participants towards realising a distributed infrastructure to evaluate and test emerging technologies, methodologies and tools related to service-based applications. It will help create new opportunities for technological research, learning and conduct experiments with service-related technologies to demonstrate research concepts and ideas have a technology underpinning. As a result, the EDSL will increase the sharing of information inside and outside the S-Cube Network of Excellence (NoE), and facilitate further collaboration among the participants.

The EDSL will provide to the different S-Cube beneficiaries two main functions: a *distributed test-bed*, i.e. a common distributed shared environment for service deployment, experimentation, and testing facilities, and a *collaboration infrastructure* to facilitate the sharing and the distribution of information within and across the S-Cube research communities by means of (for example) Web forums and file synchronisation mechanisms such as Subversion[1]. The distributed test-bed will result from the leveraging and networking of computing infrastructure provided by the S-Cube partners, and is described in Section 2.1. The collaboration infrastructure will be integrated with the S-Cube Virtual Campus (SoE-1.1) and the S-Cube Web Portal[2]. In this way researchers will be able to publish, retrieve and access the intermediate and final research outcomes produced within the project. These outcomes will include existing software prototypes, tools and documentation developed by the S-Cube partners and third-parties (hosted by the EDSL), educational resources produced for and hosted on the Virtual Campus, as well as research publications and deliverables hosted by the S-Cube Web Portal.

The functions of the EDSL and how it will support and simplify the execution of experiments by and the sharing of knowledge among the S-Cube participants is better outlined with a complex scenario, presented in Section 2.1. Section 2.2 extracts from the proposed scenario a high-level view on the requirements of the EDSL. Finally, building on top of the high-level requirements, Section 2.3 draws an overview of how the realisation of EDSL will span over the lifetime of the S-Cube project.

2.1 An Integral EDSL Scenario: Measurement of a Service Ecosystem's Responsiveness

This Section presents a use case to explain the functions provided in the long-term vision of the EDSL. Our scenario, called "Measurement of a Service Ecosystem's Responsiveness", is based on the Supply Chain Scenario for S-Cube [3], and runs as follows (letters denote phases in the scenario):

A - In order to experiment the theories and techniques developed inside S-Cube, the participants of the NoE have developed on top of the EDSL a large-scale service ecosystem that resembles the Service oriented Infrastructure described in the Supply Chain Scenario. The services ecosystem is composed of a sufficiently large number of software services, active clients, and service registries. The services are deployed on a diversity of service runtimes connected through an *Enterprise Service Bus* (ESB). The service ecosystem has been developed in the EDSL as a set of virtual machine images that contain the services and their runtime, together with the tools to monitor the services and their interactions.

B - In order to experiment with QoS and Quality Assurance, it is necessary to evaluate the responsiveness of the service ecosystem. A group of researchers from the Integration Work-Package is assigned to the task. One of the researchers sets up a project on the collaborative infrastructure. The other researchers join the project upon invitation using the integrated user management system provided by the collaborative interface. A mailing list for the project is immediately available, as well as a Web forum and a Subversion repository. Using the mailing list and the Web forums, the participants reach a consensus on the overall setup of the experiment.

C - The setup of the experiment involves the deployment and execution of software made available by the different partners, such as the services, the service clients and the services registries. Moreover, a number

[1] `http://subversion.tigris.org/`
[2] `http://www.s-cube-network.eu/`

of third-party developed applications are also needed: the runtime for the services, as a combination of deployments of Apache Axis, Microsoft WCF and Apache CXF, and an ESB, e.g. Apache ServiceMix, to connect the different service runtimes together.

The different applications that are involved in the experiment run on different operating systems, and thus must be deployed on different virtual machines, the images of which must also be configured by the participants in the project. The participants will reuse and possibly customise the set of already available virtual machine images that represent the service ecosystem and that have been previously developed. The project leader assigns tasks to the researchers in the project that are divided in research groups, using the integrated project's calendar to setup the deadlines for the different virtual machine images. A first group will prepare the services and service clients testbed generator. A second group is in charge of the service runtime. Finally, a third group is tasked with setting up two UDDI service registries and the ESB. Each research group develops locally the customised virtual machine image using virtualisation tools that can be downloaded from the software repository of the EDSL. When a virtual machine image has been configured, it is uploaded as a project file in the project's workspace.

To test the virtual machine images, a simple experiment is set up involving only the deployment of the virtual machine images and a limited interoperation of the applications deployed in them. One of the researchers is tasked with the definition of the experiment plan, i.e., the configuration file for test-experiment, which is uploaded upon completion to the Subversion repository, since the researchers plan to incrementally refine and extend it afterwards to run the full-blown experiment, which consists of a series of Monte-Carlo simulations of the service ecosystem.

D - The project leader looks up the calendar for the EDSL experiments, and schedules the test-experiment in six hours since the system is currently running other experiments and there are not enough physical resources available to support the additional amount of virtual machines to be deployed. After six hours, the test-experiment is run, and the experimental data produced (a set of graphs plotting the performance and the responsiveness of the ecosystems and logs detailing the successful completion of the deployment of the applications) are automatically posted in a dedicated area of the project's workspace. The successful outcome of the experiment is automatically emailed to the project's participants.

E - The next milestone for the project is the realisation of the experiment plan for the real experiment. The participants use the project's Subversion repository to concurrently modify and extend the experiment plan for the test-experiment into the one for the full-blown experiment. In particular, the researchers change the QoS features of the deployed services according to an agreed probability distribution and change the location and availability of some deployed services. When the configuration file is ready, the experiment is scheduled and executed.

F - Unfortunately, something goes wrong in the interaction among the applications, and the experiment fails. The applications' logs collected during the failed attempt are downloaded using the Web-based interface of the project, and examined by the researchers. It turns out that the problem is in one of the virtual images deployed, the firewall of which did not let incoming connections through. After the problem is fixed, the updated virtual machine image is uploaded again, and the experiment is scheduled for another execution.

G - This time the execution of the experiment is successful: the deployed applications work together without failures, and produce a considerable amount of data that can be accessed and elaborated using the EDSL API that gives access to the collected experimental data. However, the participants are not convinced by the outcome: one of the applications underperformed their expectations, and the participants are undecided whether it was due to a glitch in the application, or the QoS parameters that were required in the experiment plan. Thus they decide to repeat the experiment a few more times; at first using the same settings as before, and afterwards by modifying the experiment plan for stricter performance requirements. Correlating the data produced by the various iterations of the experiment, it turns out that the problem really was due to the underestimation of the required QoS parameters (in particular of the process execution time guaranteed to the application that was underperforming).

H - The new experimental results are satisfactory, and the participants in the project decide to share the final version of the experiment plan and their virtual machine images with the rest of the EDSL users; this is done by the project leader by marking the files as public in the Web-based administration interface.

The "Measurement of a Service Ecosystem's Responsiveness" scenario is broken down in phases such as "A", "B", and so on to simplify their mapping to the activities in the UML2 Activity Diagram modeling the scenario presented in Figure 1: activities in the UML diagram marked with a given letter have been modeled on the basis of the text corresponding to that letter in the use case. The activities involved in the "Measurement of a Service Ecosystem's Responsiveness" scenario are:

Create project: an EDSL user creates a project to conduct experiments. The creation of the project is carried out on the collaboration infrastructure.

Create project workspace: the workspace for the newly-created projected is automatically generated by the collaboration infrastructure.

User Management: the user that has created the project invites through the collaboration infrastructure other users to join the experiments.

Setup the experiment plan: the project's users define an experiment plan on their own, using tools not provided by the EDSL (e.g., a text editor). The experiment plan is uploaded to the workspace using the facilities of the collaboration infrastructure. The collaboration infrastructure simplifies the concurrent definition of the experiment plans by allowing multiple users to share their work using versioning mechanisms such as SVN.

Assign task to participants: project users can define, assign and monitor the completion of tasks using the facilities provided by the collaboration infrastructure.

Create virtual machine images: the project's users define virtual machine images using tools (not provided by the EDSL, such as VMWare Player). Once the virtual machine images are ready, they are uploaded to the workspace using the facilities of the collaboration infrastructure.

Configure experiment: the users use the collaboration infrastructure to prepare an experiment plan, which is then uploaded to the workspace to be run.

Schedule experiment: the users book on the test-bed the resources required to execute the experiment through the calendar system provided by the workspace.

Execute experiment: on the scheduled time, the test-bed executes the experiment, collects the experimental data, and upon the completion of the execution uploads the results to the project workspace.

Analyze experimental results: the users access the experimental data produced during the execution of the experiment, and use tools (e.g., Gnuplot, integrated in the collaborative infrastructure) to analyze the experimental results.

Change Experiment Plan: modify an already existing experiment plan to improve it, or correct defects. Similarly to the case of the **Setup the experiment plan** activity, the collaboration infrastructure supports the concurrent work on the experiment plans by the users in the project.

Share results: the experimental data, the experiment plan and the virtual machine images used to run the experiment are shared with the other projects on the EDSL.

2.2 High-level View on the EDSL Requirements

From the "Measurement of a Service Ecosystem's Responsiveness" scenario, it becomes obvious that the EDSL provides an added value to the S-Cube beneficiaries, enabling the correct setup of shared experiments and by allowing an efficient dissemination of knowledge and experimental data. Furthermore it can promote collaboration and reuse of research results, and the validation of research findings.

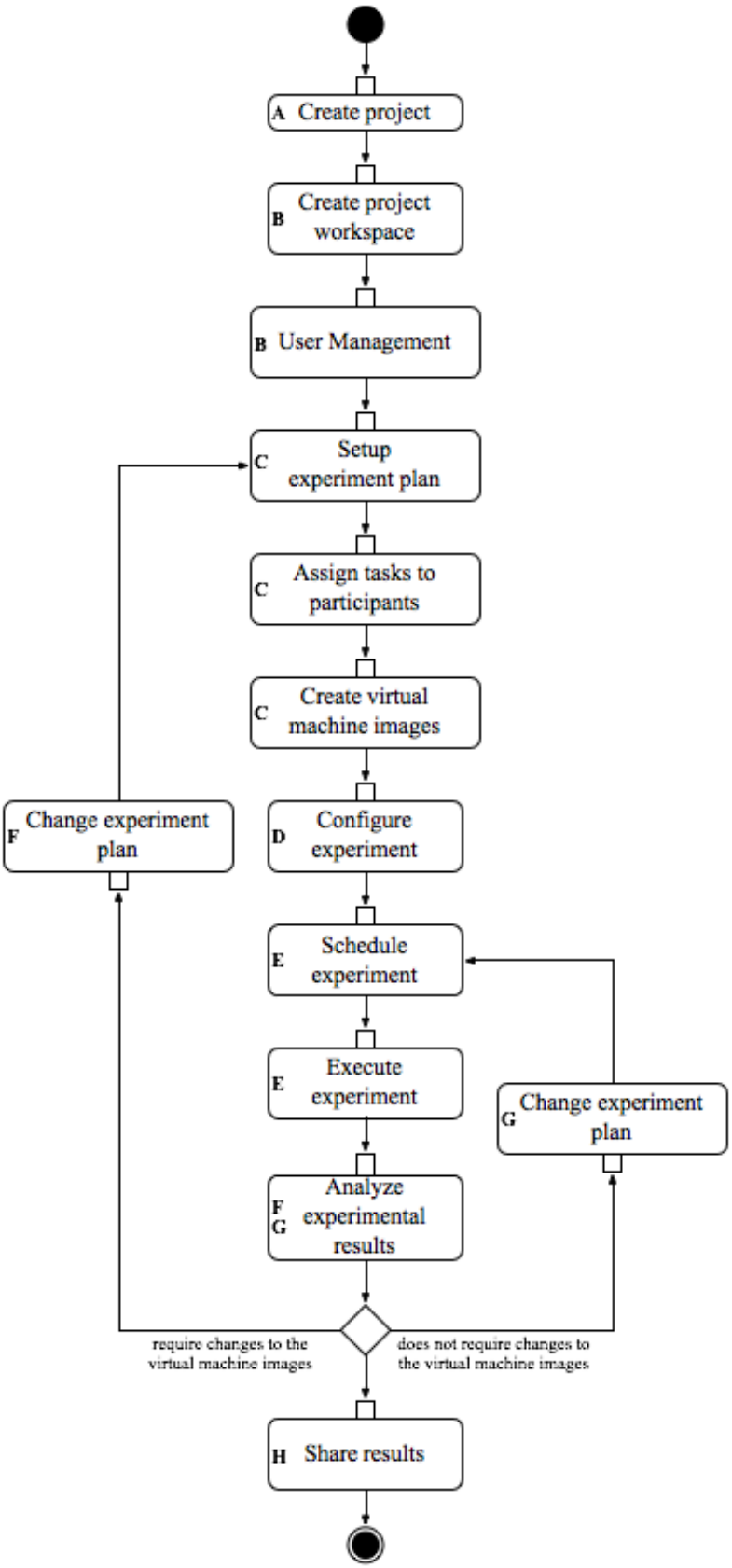

Figure 1: A UML2 Activity Diagram modeling the "Measurement of a Service Ecosystem's Responsiveness" scenario.

This Section provides a high-level view on the requirements that can be found in the "Measurement of a Service Ecosystem's Responsiveness" scenario. These high-level requirements constitute the baseline for the more detailed definition of the EDSL requirements presented in Section 3.

2.2.1 Collaboration Facilities

The EDSL provides a collaboration infrastructure that allows users to setup projects that support the execution of experiments, the shared development of demonstrators and prototypes, and more generally the collaborative realization of research artifacts (e.g. papers, deliverables, etc). The projects are associated with workspaces. The workspaces are inter-connected environment, accessible through a web-based interface, in which all the participants in geographically dispersed locations can access and interact with each other. The workspace provides an array of collaboration functions such as document management to share files within the project, file versioning systems, collaborative knowledge bases (i.e., Wikis), mailing lists, forums and calendar.

Users can autonomously create new projects on the collaborative interface. After the creation of a project by a user, she can invite other users to join the project and access the workspace. Project users may be granted access to management functionalities such as role-based access controls for permission setting of project members and project outsiders. The entire workspace or portions of it (e.g. the forum, the versioning system, the calendar, etc) may be set to private to restrict access to project members only (i.e., project outsiders can not access private portions of the project's workspace).

Project users can execute experiments on the distributed test-bed (see Section 2.2.2). The workspace supports the preparation scheduling of experiment executions (see Section 2.2.3) through its calendar system, and the collection and sharing of experimental results (see Section 2.2.4) through its document management facilities.

2.2.2 Distributed Test-Bed

The EDSL provides its users with a distributed computing platform, called *distributed test-bed*, that employs virtualization and Grid technologies to support a unified access to the deployment, instantiation, migration, cloning and decommissioning of *virtual machines* (i.e., independent instances of operative systems that are isolated from the physical hardware by the virtualization platform such as Xen Source[3] and VMWare Server[4]) on which the users can deploy and run software to execute experiments (see Section 2.2.3).

Machines can be contributed to or removed from the distributed test-bed by the S-Cube partners. Users can book the usage of resources available to the distributed test-bed through the experiment scheduling facilities provided by the EDSL's collaboration facilities (see Section 2.2.1). The distributed test-bed collects detailed logs about its workload and how it is allocated by the different projects, the management of physical and virtual machines, and more generally information necessary to track the usage of the EDSL.

2.2.3 Planning and Execution of Experiments on the EDSL

The EDSL provides advanced functions to plan and execute automated experiments. Each experiment is treated as a different project in the collaboration infrastructure (i.e., each experiment has a dedicated workspace, see Section 2.2.1). Project members collaboratively define one or more *experiment plans*, which describe the actions that the EDSL must take in order to run the experiment, such as which virtual machine images to deploy on the distributed test-bed, how long each is the experiment expected to last (upper bound to prevent lockups of distributed test-bed resources), conditions of success and failure for the experiment, how the data produced during the experiment have to be collected, elaborated and shared

[3] `http://www.xensource.com/`
[4] `http://www.vmware.com/products/server/`

in the workspace (see Section 2.2.4), how the project members will be notified during and after the execution (e.g. via email, with an outlook of the current data collected and logs from the virtual machine images), etc.

The execution of experiments on the distributed test-bed is scheduled by the users through the collaboration functions provided by the project's workspace (i.e., integrated calendar, see Section 2.2.1), compatibly with the availability of test-bed's resources. Experiment plans are reusable (i.e., the same experiment plan can be scheduled multiple times). When an experiment is run, a distributed experimental test-bed is automatically set up, linking resources geographically distributed across the partners and infrastructures as specified in the experiment plan. The virtual machine images containing the service runtimes are automatically deployed on the physical machines in the test-bed. The data produced by the experiment execution are stored in databases and published on the collaboration infrastructure according to the specifications of the experiment plan (see Section 2.2.4). When the experiment is completed, the resources are released (e.g. the virtual machines are stopped and undeployed), and made ready to be used again in the next experiment.

2.2.4 Storing, Elaborating, and Sharing Experiment Results

The EDSL provides an infrastructure for collecting and processing *experimental data* produced during the execution of experiments (outlined in Section 2.2.3). The experimental data are divided in *raw data* (produced by the software running on the test-bed), and *processed data* (e.g. graphs, views on databases), which can be produced during or after the execution of the experiment as specified by the experiment plan.

The raw data are produced during the experiment by the software and services deployed in the virtual machines that invoke operations provided an API in a way similar to logging APIs à la *Apache Log4J*[5]. The raw data are stored into relational repositories in the collaboration infrastructure of the EDSL, and can be queried via SQL by the researcher for a limited amount of time after the completion of the experiment itself. The experiment plan allows to specify how the raw data have to be elaborated into processed data (e.g. visual formatting of graphs using graphical commands[6], database views defined in SQL). The experimental data can be accessed at run-time during the experiment.

As soon as the experiment is completed, a copy of the experimental data is stored into one or more files (according to the kind and size of the data) that are published in the experiment's workspace for further access by the project members. The project members can use the file-management functions of the workspace to modify the experimental data after they have been published.

2.2.5 Integration between the EDSL and other S-Cube Information Systems

Four main information systems are to be developed inside S-Cube: the S-Cube Web Portal, the EDSL, the Virtual Campus and the Knowledge Model. This Section outlines the integration of the EDSL with the other S-Cube information systems.

EDSL and the S-Cube Web Portal

The S-Cube Web Portal is a web site that publishes the results produced by the S-Cube project, news and events, job offers, and information about the S-Cube consortium. The Web Portal has a private working area, accessible only to the S-Cube researchers, where all the researchers can post document and resources related with S-Cube (usually related with the development of a deliverable).

Currently, there are no plans to integrate the infrastructure of the Web Portal (Apache web server and plone as content management system) with the infrastructure of the EDSL. However, dedicated links / dependencies between the information hosted by the two information systems will be established:

[5] `http://logging.apache.org/log4j/`

[6] For instance, *Gnuplot* commands: `http://www.gnuplot.info/`

- The Web Portal will provide access to the laboratory facilities by linking to the EDSL Portal.
- Secondly, while the working area of the Web Portal will host major releases of papers, deliverables, presentations and other documents, the collaboration functions of the EDSL (e.g. file synchronization and version control) will be used to enable the joint work on documents and allow for fine-grained version control.

EDSL and the Virtual Campus

The Virtual Campus (to be specified and developed in the SoE-1.1 Work-package) [2] is meant to be a virtual community network through which researchers involved S-Cube and, more generally in the SOC/SOA community can communicate and have discussion forums and be informed of new research, methodologies and development activities, including business cases and training opportunities.

The experimental results obtained through the EDSL can be integrated in the didactic offer of the Virtual Campus. For this purpose, the EDSL will provide a "1-click" function to publish experimental data on the Virtual Campus.

EDSL and the Knowledge Model

The Knowledge Model [1] is developed within the work package WP-IA-1.1 with purpose of "synthesizing and integrate diversified knowledge", thereby fulfilling the mission of innovation and integration of research agendas of disparate research groups in software services. The S-Cube Knowledge Model is the central repository of terminological definitions for the NoE, and it will be used by the EDSL in two ways:

1. As a reference point for matters of terminology in the definition and analysis of the experiments.
2. As a knowledge source in identifying the necessary expertise and possible collaborations within and across the NoE for organizing, conducting and analyzing the results of experiments.

2.3 Medium- and Long-Term Visions for the EDSL

Realizing the requirements of the EDSL is not trivial. The EDSL must support the experimentation with technologies currently available, and that may become available in the future. Thus, it has to be flexible and versatile; moreover, it has also to be robust, resilient and reliable.

Developing software solutions is outside the scope of S-Cube as a NoE. Thus, in order to realize the EDSL, we foresee an incremental integration of existing software solutions such as virtualization infrastructures and collaboration frameworks. The integration with the Virtual Campus and the S-Cube Web Portal is important for providing the end-users of the three S-Cube's component systems a consistent and comprehensive experience, and truly fulfill the potential of leveraging an experimentation infrastructure such as the EDSL with the educational and research contents delivered by the Virtual Campus and the S-Cube Web Portal. The creation of the distributed test-bed, and its integration with the collaboration facilities is a complex task, that requires development effort, and that can be potentially carried out at different levels of completeness and refinement. Hence EDSL will be realized reusing as much freely available (possibly open-sourced) technology, while focusing the coding and customization efforts on key areas (such as the ease of use, and a lightweight infrastructure to avoid serious development efforts).

The realization of the EDSL is split in two segments, called *medium-* and *long-term vision*, respectively described in Section 2.3.1 and Section 2.3.2. The remainder of this Section sketches the medium- and long-term vision; a more detailed timeline for the EDSL is presented in Section 4.

2.3.1 Medium-Term Vision

The medium-term vision for the realization of the EDSL aims at achieving the first prototyping (Month 23, deliverable CD-IA-1.2.4) and public rollout (Month 39, deliverable CD-IA-1.2.6) with a bulk of functions that will render the EDSL usable by users with a technical background. The realization of the medium-term vision requires the deployment and configuration of well-established software packages and platforms. The mailing lists, Web forums, calendar system, sharing of file across projects and versioning systems associated with the project are core functions of the collaborative infrastructure, as well as sharing files among projects and with the whole EDSL user base. Similarly, the possibility of deploying the virtual machine images that interact with each other is a core feature of the virtualization infrastructure.

While the scenario presented in Section 2.1 has been drawn with the long-term vision for the EDSL in mind, most of it can also be fulfilled by the medium term vision. Nonetheless, the medium-term vision already grants to its users a consistent part of the added value of the EDSL. In particular, the medium-term vision offers everything described in the use case except for the automation of the experiment execution and the automatic collection of the experimental data. The EDSL users will manually configure, deploy and manage virtual machines interacting with the system at low-level using the communication and Grid infrastructures, and manage shared resources through the collaboration infrastructure. The integrated functions such as the collection and elaboration of the experimental data and their posting on the collaboration infrastructure are instead part of the long-term EDSL vision, as well as the automation of experiment execution.

2.3.2 Long-Term Vision

EDSL's more advanced features will be gradually and incrementally realized over the lifespan of the project. The long-term vision foresees an EDSL where composite, integrated, value-added functions are built on top of the low-level ones provided by the realization of the medium-term vision. The functions comprised in the long-term view mainly focus on leveraging the different technologies of the EDSL, and their realization involve some software development effort to customize the deployed software and to realize the connections. The fulfillment of the long-term vision can outlive the span of the S-Cube project, and it will commence with the definition of the sustainability plans for the EDSL outreach to industry and small and medium enterprises (deliverable CD-IA-1.2.7, due in Month 48). The requirements that will be fulfilled in the long term are not limited to the ones presented in this deliverable. They will be further refined during the implementation and deployment of the EDSL, benefiting from the experience accumulated by its users.

3 Requirements for the Pan-European Distributed Services Laboratory

This Section outlines the requirements from the EDSL in a more detailed and structured way than what is been presented in Section 2.2.

This Section is structured as follows: Section 3.1 outlines the actors and stakeholders of the EDSL. Section 3.2 lays down the requirements for the EDSL. Finally, Section 3.3 maps the requirements to the medium- and long-term vision for the EDSL.

3.1 EDSL Actors and Stakeholders

The EDSL stakeholders represent any person or group who will be affected by the EDSL, directly or indirectly. They include end-users such as S-Cube researchers, who interact with the systems, executive stakeholders such as the European Commission and the S-Cube steering committee, who evaluate the the impact and the usefulness of the EDSL on the S-Cube project, and technical experts in charge of installing, configuring and maintaining the EDSL at the partner sites.

The different typologies of EDSL users span from S-Cube researchers, JRA leaders, Activity leaders, PhD candidates and partners' students to technicians (e.g. system administrators at some S-Cube partner's facility) and EDSL managers.Users can play different roles when interacting with the EDSL:

Administrator: the administrator is responsible for the overall management of the hardware and software resources involved in the EDSL. Given the collaborative and distributed facets of the EDSL, this role is actually split in two separate ones:

- **Collaborative Infrastructure Administrator:** administers the collaborative infrastructure.
- **Distributed Test-Bed Administrator:** administers the physical resources constituting the distributed test-bed that are hosted in the partners research facilities, and the layer of middleware that constitutes the distributed test-bed.

Supervisor: collects data about EDSL usage and contents and to monitor key performance indicators to assess the usage and the usefulness of the EDSL.

User: similarly with the case of the Administrator, due to the duality of the EDSL there are two different roles played by the users:

- **Collaborative Infrastructure User:** interacts with the collaborative infrastructure, such as create and join projects. A Collaborative Infrastructure User can assume one of the four following sub-roles with respect to a particular project on the EDSL's collaboration infrastructure (see also the Requirement **Project Management**):
  - **Project Leader:** the Collaborative Infrastructure User that creates the project. She can access the full extent of the project's management facilities (e.g. sharing of resources, user management, etc).
  - **Project Deputy:** a user that has joined the project and to whom the Project Leader has granted access to the project's management facilities.
  - **Project Member:** a user that has joined the project but that does not have access to any of the project's management facilities.
  - **Project Outsider:** a Collaborative Infrastructure User that has not joined the project.
- **Distributed Test-Bed User:** logs in to the distributed test-bed to manually perform the deployment of virtual machine images and direct interaction with them.

There are two collective roles defined on projects on the collaborative interface (see also the Requirement **Project Management**):

Project Board: the set comprising the Project Leader and all the Project Deputies.

Project Staff: the set comprising the Project Leader, the Project Deputies and the Project Member.

The same physical person might play multiple roles on the EDSL. For instance, a technician at some partner who play the role of Distributed Test-Bed Administrator for that partner's facility might also be a Collaborative Infrastructure User (to download useful software from the EDSL) and a Distributed Test-Bed User (for instance for testing purposes).

3.2 EDSL Requirements

User Management: every USER must register an account. An ADMINISTRATOR will need to acknowledge each registration before the USER can access the EDSL. The EDSL will respond to a registration with an e-mail which demands the USER to follow a link back to EDSL to verify the e-mail address. The ADMINISTRATOR can give users access to projects and privileges on repositories, individual files, etc. The account information (username, password, personal details) may be shared with the Virtual Campus and/or the Web Portal. USERS must authenticate themselves before accessing the EDSL facilities. Once a USER has been authenticated and until she logs out from the EDSL, no more authentication should be required to perform any action on the EDSL (i.e. single sign on).

Project Management: The management control over a workspace is granted to the PROJECT LEADER. All the COLLABORATIVE INFRASTRUCTURE USERS that are not project members of a particular project are collectively known as PROJECT OUTSIDERS for that particular project. The PROJECT LEADER can grant access to the management facilities of the workspace to other PROJECT MEMBERS, who become PROJECT DEPUTIES. The PROJECT LEADER and the PROJECT DEPUTIES are collectively referred to as PROJECT BOARD. PROJECT OUTSIDERS that want to access a project and its workspace have to be invited by a member of the PROJECT BOARD, and they become PROJECT MEMBERS as soon as they accept the invitation. Alternatively, PROJECT OUTSIDERS can apply to join the project, and their request has to be accepted by a member of the PROJECT BOARD; when the application for project membership is accepted, the PROJECT OUTSIDER becomes a PROJECT MEMBER. The project board and all the PROJECT MEMBERS are collectively known as PROJECT STAFF. Members of the PROJECT BOARD have the customary user management capabilities of collaborations systems (e.g. add, remove, ban PROJECT STAFF). The PROJECT BOARD has fine-grained role-based access controls for permission setting of PROJECT STAFF and PROJECT OUTSIDERS. The entire workspace or portions of it (e.g. the forum, the versioning system, the calendar, etc) may be set to private to restrict access to PROJECT STAFF only (i.e. PROJECT OUTSIDERS can not access private portions of the project workspace).

Document Management: PROJECT STAFF can upload, download and delete files posted on the web-based file repository, à la S-Cube Web Portal Working Area [7].

File Synchronisation and Version Control: PROJECT STAFF are provided with a centralised versioning systems e.g. Subversion.

Collaborative Knowledge Base: each project automatically hosts a Wiki that can be accessed and modified by the PROJECT STAFF. Typical wiki's moderation mechanisms are available to the PROJECT BOARD.

Mailing Lists: the workspace can automatically host any number of mailing lists. PROJECT STAFF can join and leave the mailing list at will. Also, PROJECT BOARD can add and remove users from mailing lists.

[7] http://www.s-cube-network.eu/working-area

Surveys and Polls: PROJECT STAFF can submit topics for and vote in polls. Typical survey moderation mechanisms are available to the PROJECT BOARD.

Task Management: the PROJECT BOARD can assign *tasks* and *todos* (tasks without a precise deadline) to the PROJECT STAFF. Tasks can be visualised in lists (ordered by a diversity of criteria like task's deadline, task's name, etc) or in the shape of Gantt charts. PROJECT STAFF in charge of a task can update its progression, and write and edit notes describing the current status of the task.

Calendar System: PROJECT STAFF can enter entries in the *project calendar*. The scheduled experiments appear also in the project calendar.

Distributed Test-Bed: the EDSL provides a distributed test-bed that employs virtualisation and Grid technologies to support a unified access to the deployment, instantiating, migration, cloning and decommissioning of *virtual machines*, i.e. independent instances of operative systems that are isolated from the physical hardware by the virtualisation platform (e.g. Xen Source, VMWare Server, etc).

Physical Resources Management: DISTRIBUTED TEST-BED ADMINISTRATOR can add and remove physical machines the distributed test-bed. These operations may require specific software installation/configuration and synchronisation with other physical resources.

Virtual Resources Management: DISTRIBUTED TEST-BED ADMINISTRATOR can deploy virtual machines to and undeploy them from the distributed test-bed.

Resource Reservation: PROJECT STAFF can book the usage of resources available from the distributed test-bed to execute experiment defined in their projects.

Experiment Plan Management: each experiment is treated as a different project in the collaboration infrastructure (i.e. each experiment has a dedicated workspace). PROJECT STAFF collaboratively define one or more *experiment plans*, which describe the actions that the distributed test-bed must take in order to run the experiment:

- amount of instances of certain virtual machines to be deployed on those particular nodes;
- how long each is the experiment expected to last (upper bound to prevent lockups of distributed test-bed resources);
- conditions of success and failure for the experiment;
- how the data produced during the experiment have to be collected, elaborated and shared in the workspace;
- how the project members will be notified during and after the execution (e.g. via email, with an outlook of the current data collected and logs from the virtual machine images, etc.).

The PROJECT BOARD can schedule experiment plans to be executed on which date, compatibly with the availability of resources of the distributed test-bed and an estimation of the length of the experiment provided in the experiment plan. Experiment plans are reusable (i.e. the same experiment plan can be scheduled multiple times).

Automated Experiment Execution: When an experiment is run, the distributed experimental test-bed is automatically configured, linking resources geographically distributed across the partners and infrastructures as specified in the experiment plan. The virtual machine images containing the service runtimes are automatically deployed on the physical machines in the test-bed. The data produced by the experiment execution are stored in databases and published on the collaboration infrastructure according to the specifications of the experiment plan. When the experiment is

completed, the resources are released (e.g. the virtual machines are stopped and undeployed), and made ready to be used again in the next experiment. Executions of experiments can possibly span over long periods of time. According to the setup of the experiment plan, PROJECT STAFF can visualise the experiment results in real time, as well as receive notification via mail of the outcome of each experiment execution as soon as it is completed.

Experiment Results Management: the data collected during the experiment are called *raw data* to differentiate them from possibly elaborated data, called *processed data*, that may be produced during or after the execution of the experiment as per the experiment plan. Raw and processed data are collectively known as *experimental data*.

The raw data are produced during the experiment by the software and services deployed in the virtual machines that invoke operations provided an API in a way similar to logging APIs. The raw data are stored into repositories in the collaboration infrastructure of the EDSL. The raw data produced by an experiment may be queried by the PROJECT STAFF for a limited amount of time after the completion of the experiment itself. The experiment plan allows to specify how the raw data have to be elaborated into processed data (e.g. visual formatting of graphs using graphical commands). As soon as the experiment is completed, a copy of the experimental data is stored into one or more files (according to the kind and size of the data) that are published in the project workspace for further access by the PROJECT STAFF. PROJECT STAFF can use the file-management functions of the workspace to modify the experimental data after they have been published.

Integration with Virtual Campus: the EDSL and the Virtual Campus share the same user management system. A simple, one-click procedure must be provided by the EDSL to publish experimental data and/or results on the Virtual Campus. A similarly simple procedure is provided to share software hosted/developed on the EDSL's collaborative infrastructure.

Integration with S-Cube Web Portal: the EDSL and the S-Cube Web Portal share the same user management system.

Integration with Knowledge Model: the EDSL and the Knowledge Model share the same user management system. PROJECT STAFF can inquire the Knowledge Model from inside the EDSL to find out PROJECT OUTSIDERS whose expertise may be beneficial to the project. The EDSL displays keywords or definitions with a corresponding entry in the Knowledge Map, a link to that entry is provided.

Utilisation Reports: SUPERVISORS and ADMINISTRATORS can access *utilisation reports* that resume the usage of the EDSL in a given span of time (which might cover the entire life of the EDSL). A utilisation report include the following data (list not exhaustive, to be extended later on in the development of the EDSL):

- Number of projects
- Breakdown projects: for each project, the following data are listed
  - PROJECT LEADERS
  - PROJECT DEPUTIES
  - PROJECT MEMBERS
  - Number of experiments run (and average execution time)
  - Number of virtual machines deployed
  - Average lifetime of a virtual machine
  - Mailing list and forum activity (e.g. average of messages per day)
  - History of the releases of files

Requirement	Medium-term Vision	Long-term Vision
User Management	✓	
Project Management	✓	
Document Management	✓	
File Synchronisation and Version Control	✓	
Collaborative Knowledge Base	✓	
Mailing Lists	✓	
Surveys and Polls	✓	
Task Management	✓	
Calendar System	✓	
Distributed Test-Bed	✓	
Physical Resources Management	✓	
Virtual Resources Management	✓	
Resource Reservation	✓	
Experiment Plan Management		✓
Automated Experiment Execution		✓
Experiment Results Management		✓
Integration with Virtual Campus		✓
Integration with S-Cube Web Portal		✓
Integration with Knowledge Model		✓
Utilisation Reports	✓	

Table 1: The partition of the requirements specified in Section 3.2 with respect to the medium- and long-term vision.

3.3 Mapping the EDSL Requirements to Medium- and Long-Term Visions

The medium- and long-term visions for the EDSL, introduced in Section 2.3, represent two different stages of development for the EDSL. The medium-term vision, outlined in Section 2.3.1, foresees an EDSL that, even if lacking the more refined features such as the automation of the experiments execution, will provide functions that will render it usable by users with a technical background. The long-term vision, instead, foresees a "full-blown" EDSL providing automation of experiments execution, the integration with the other S-Cube information systems, etc.

In order to define precisely which EDSL functions are part of either of the visions, Table 1 maps the requirements expressed in Section 3.2 to the medium- and long-term vision. The requirements are correlated to the two visions. A checkmark (✓) means that the respective requirement will be realised in the vision corresponding to the column.The mapping has been devised on the basis of the descriptions of the two visions as respectively described in Section 2.3.1 and Section 2.3.2.

4 The EDSL Timeline

The realization of the EDSL will span a period of more than three years, starting from the beginning of the work on the deliverable PO-IA-1.2.3 (Plan for the organizational and operational structure of the EDSL) at Month 10 to the completion of the S-Cube project at Month 48 and possibly beyond. Figure 2 presents a Gantt chart outlining the timeline for the deliverables in the IA-1.2 Work-Package (represented by the Tasks 1 7) and relating them with a high-level overview of the EDSLs development process (modeled by the Tasks 8 10). Planning for the extension of the lifetime of the EDSL beyond S-Cube is the goal of the deliverable CD-IA-1.2.7 (Month 48), which will define sustainability plans to involve industrial partners and SMEs as committed EDSL beneficiaries to allow self-sustainability of the laboratory after the S-Cube founding ceases. The implementation plan for the EDSL is divided in three segments, the first two of them focusing on the medium-term vision, and the third one embracing the long-term vision:

Development of the EDSL pilot implementation (Figure 2, Task 8): a first implementation, called in the remainder *pilot implementation*, is due to Month 23 (deliverable CD-IA-1.2.4). Notice that, in the description of the deliverable CD-IA-1.2.4, the pilot implementation is referred to as "First Prototype." We prefer to use the term pilot implementation in this timeline to underline the fact that S-Cube participants will work on it and that it will gradually evolve into the production system. On the one hand, in Software Engineering, the term "prototype" usually refers to a model of the target system to demonstrate concepts and functions to be developed. In other words, prototypes are usually "reality checks" before building the actual system. Prototypes are never production environments, as they are usually discarded at some point of the software development process. On the other hand, a pilot implementation is a first version of a system that is released for use, although possibly to a restricted number of users and sites, and may have some functional modules missing. A pilot implementation can be upgraded in due time by adding more features, until the mature/production ready state is reached.

The pilot implementation will offer functionalities foreseen by the medium-term vision of the EDSL (but possibly not all of them). At this point the EDSL will not be expected to ready for production use, i.e. to be stable and expose the quality characteristics expected by a software product ready to be used in business. The planning for the development and consolidation of the pilot implementation will be the main focus of the upcoming PO-IA-1.2.3 deliverable (Month 15). The development of the pilot implementation of the EDSL foresees the deployment and initial integration of a diversity of technologies ranging from Grid to Virtualization to Collaboration. As outlined in the previous deliverable PO-IA-1.2.1, these technologies can be provided by available open-source (or otherwise freely accessible) software solutions that can be deployed independently, each one providing a subset of the medium-term EDSL functionalities. Initially, the deployed systems will not be integrated with each other (as their integration is mainly an objective of the long-term vision). Nonetheless, they will require customization and configuration in order to provide a solid foundation for the later development of the EDSL.

Consolidation of the pilot implementation (Figure 2, Task 9): the goal is to iteratively evolve the pilot implementation into a system ready for production use implementing all the functionalities of the medium-term vision. The refinement of the EDSL pilot implementation will focus on addressing the feedback (bug reporting, request for tuning of features and infrastructures) that will come from the EDSL user base, and optimizing the system to meet the quality and performance requirements specified for the public rollout. Tools like TRAC[8] (a Web-based bug-tracking system) will be deployed to allow EDSL users to suggest improvements to the available functionalities, notify glitches in the system, and more generally provide feedback. The end of the consolidation process will coincide with the public rollout due to Month 39 (deliverable CD-IA-1.2.6).

[8] `http://trac.edgewall.org/`

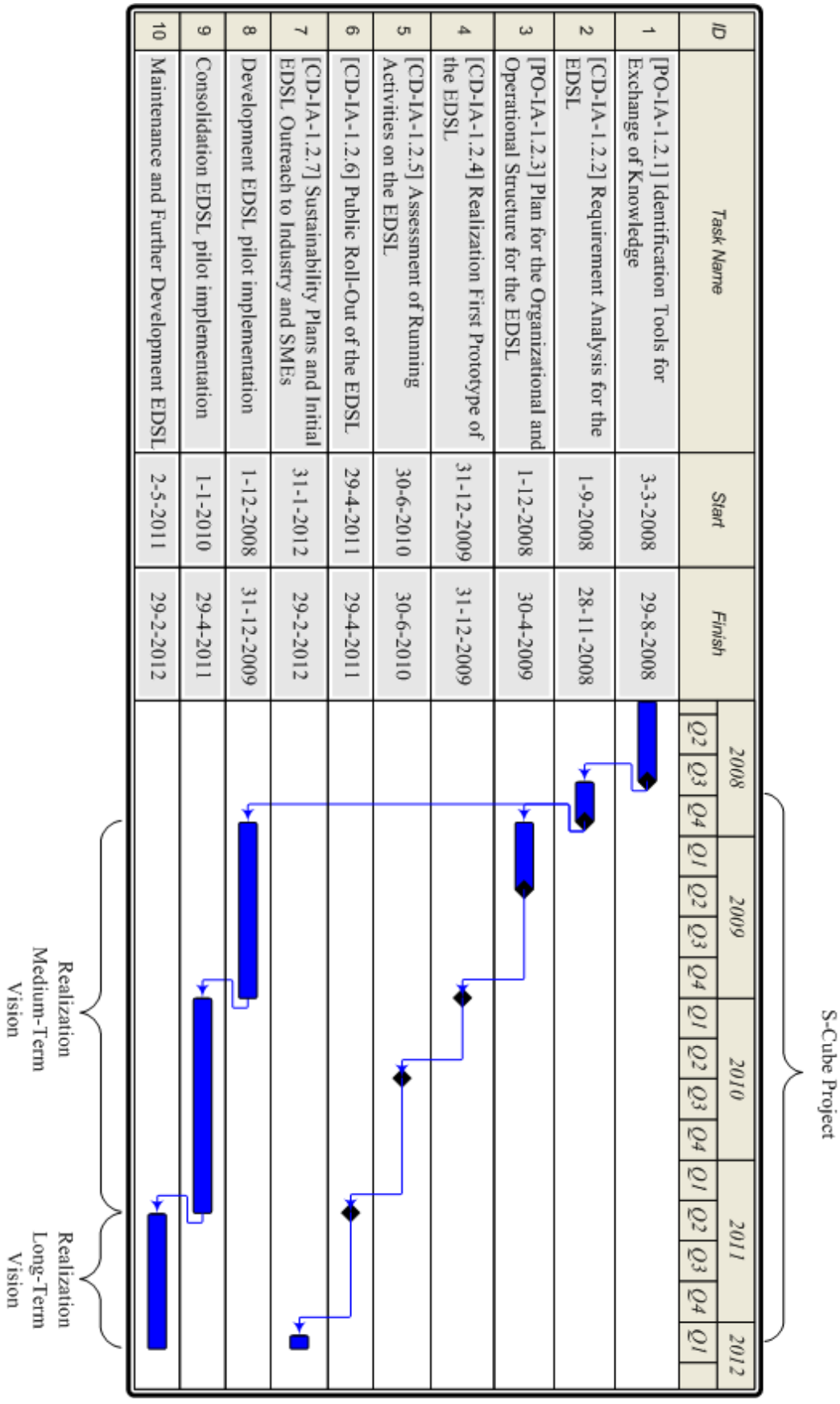

Figure 2: A Gantt chart of the high-level view on the EDSL development and maintenance process.

Maintenance and further development of the EDSL (Figure 2, Task 10): at this stage the EDSL is made available to a broader community (i.e. open to users in private and public organizations externals to S-Cube). The ongoing the development process will comprise the maintenance of the system in its current state (e.g. more bug fixing in functionalities already available), and the planning and development of the functionalities foresee in the long-term vision. The actual development plans for the realization of the long-term vision will be based on the sustainability plans outlined in the deliverable CD-IA-1.2.7, and they will ultimately depend on the development resources and needs of the partners that will contribute to the post-S-Cube life of the EDSL.

5 Summary and Future Work

The Pan-European Distributed Service Laboratory (EDSL) is a joint effort from the S-Cube partners to realise a shared, versatile and flexible infrastructure to test research results, achieve integration of research, conduct experiments with service-related technologies and demonstrate that research concepts and ideas have a technology underpinning, increase the sharing of information inside and outside the network of excellence, and facilitate further collaboration among the participants. The EDSL will provide a common distributed shared environment for service deployment, experimentation, and testing facilities to the different S-Cube beneficiaries to supporting the sharing and the distribution of information within and across the S-Cube research communities through a common Web-based infrastructure.

The present deliverable presented the vision of the S-Cube partners for the EDSL in terms of the infrastructures necessary to realise it, the features it will provide to its users, and how these features will be realised and provided over the EDSL life-span. The evolution of the EDSL is split in to two parts: the medium- and long-term vision. The medium-term vision will provide the EDSL users with the singular functionalities provided by the different technologies deployed, and will provide a limited amount of integration among them. The long-term vision will focus on higher-level, value-added functionalities resulting from the integration the lower-level ones provided by the realisation of the medium-term vision. On top of the vision for the EDSL, this deliverable outlined requirements extracted from it, and a timeline that sketches the following steps in the IA-1.2 Work-Package.

References

[1] Vasilios Andrikopoulos, Alea Fairchild, Willem-Jan van den Heuvel, Raman Kazhamiakin, Philipp Leitner, Andreas Metzger, Zsolt Nemeth, Elisabetta di Nitto, Mike P. Papazoglou, Barbara Pernici, and Branimir Wetzstein. S-Cube Deliverable PO-IA-1.1.1: Comprehensive overview of the state of the art on service-based systems. Deliverable, S-CUBE Consortium, September 2008.

[2] Olivier Barais, Marina Bitsaki, Olha Danylevych, Schahram Dustdar, Mohand-Said Hacid, Christos Nikolaou, Frank Leymann, Elisabetta Di Nitto, Dimka Karastoyanova, Raman Kazhamiakin, Philipp Leitner, Barbara Pernici, and Ita Richardson. S-Cube Deliverable CD-SoE-1.1.2: Organisational structure for virtual campus. Deliverable, S-CUBE Consortium, March 2009.

[3] Dinh Khoa Nguyen. Supply chain scenario for S-Cube. Technical report, S-CUBE Consortium, February 2009. http://www.s-cube-network.eu/working-area/activities-and-workpackages/ia-3/research-framework/scenarios/service-network-in-automotive-industry.

[4] Ian Sommerville. *Software Engineering*. Addison Wesley, 8th edition, May 2006.