D-Grid

InGrid

Innovative Grid-Entwicklungen für Ingenieurwissenschaftliche Anwendungen

AP 4.2: Kooperations- und Geschäftsmodelle

D 4.2.1
Betrachtung typischer Anwendungsszenarien und Vorschläge für Kooperationsmodelle

Table of Contents

1. Overview

This report collects materials on business models that were elaborated in the context of the InGrid project, mainly in workpackage 4.2. The work done was partially performed on the base of pre-existing know-how that the partners have acquired either in other grid-projects like UniGrids, NextGrid, TrustCom and others, or as part of their daily business. Therefore, this survey gives a comprehensive overview about the grid-business in science and industry from some of the major European players in the field.
The report analyzes the experiences from the past and on this base, derives new business- and collaboration models that the partners intend to realize in the near future.

2. Business- and Cooperation Models in Technical Computing

2.1 Grid-Provisioning

The provisioning of grid-middleware in the environment of public customers as well as industrial research and development customers is the most elementary grid-service that can be offered. It has the aim to give a stable base for solutions based on grid-technology even on the basis of software-components that are still under development and which come from the open-source sector.

The focus is on a seamless integration of such innovative environments into a modern service and process landscape.

2.1.1 Service-Architecture

The service-architecture for the provisioning of grid-middleware strictly follows the best-practice rules and processes described in the IT Infrastructure Library (ITIL) according to British Standard 15000.

In the following diagram the interaction of users, Service-Desk, Developers and Service-Management is described. This interaction forms the base for the ITIL-processes:

- Incident-Management
- Problem-Management
- Change-Management
- Release-Management
- Service-Level Management

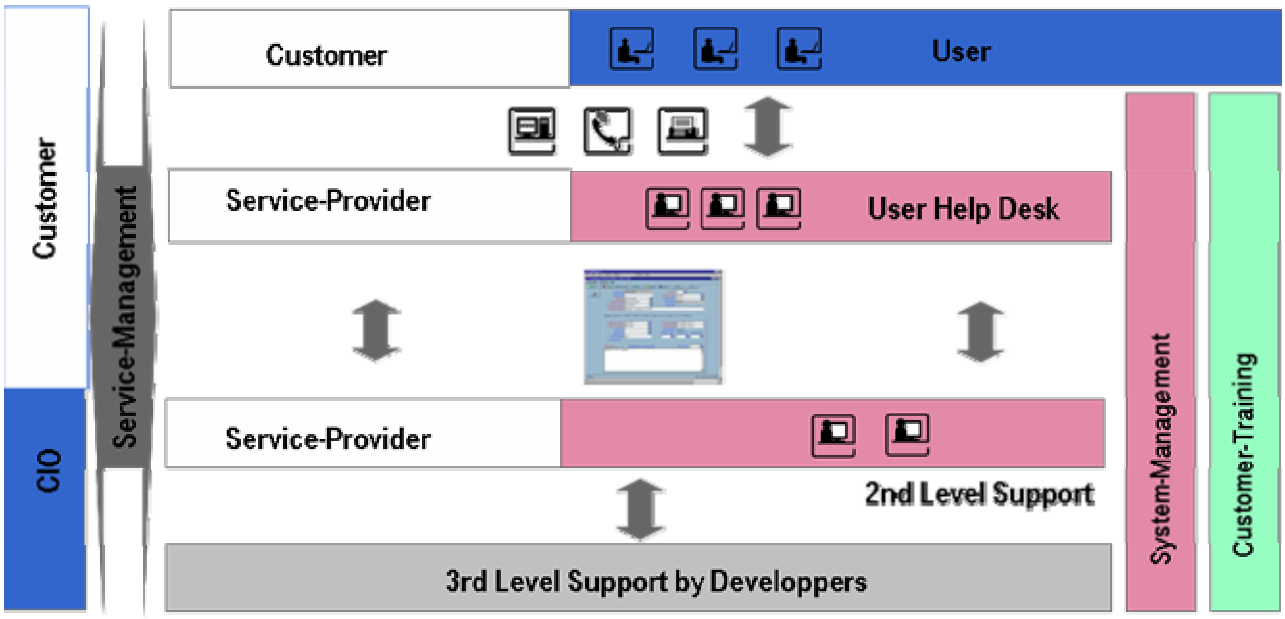

Figure 4: Grid-Provisioning

2.1.2 Release-Management

Major goal of the Release-Management for open source software is the continuous integration of new developments from related projects into production-environments. As customers expect proven and stable software products, the results of the development work performed in projects need to be checked and tested thoroughly before the software gets installed at a customer site. A second goal for complex software is that code has to be managed in a reproducible way. The rules of Release Management describe how this process of software integration should be formalized and automated.

The needs of Release Management concerning the test and quality-assurance of a software product are fulfilled by Continuous Integration Tools (CIT). After the evaluation of several CITs, the open source product Gump from the Apache Software Federation was selected and adapted to the requirements of the Release-Management.

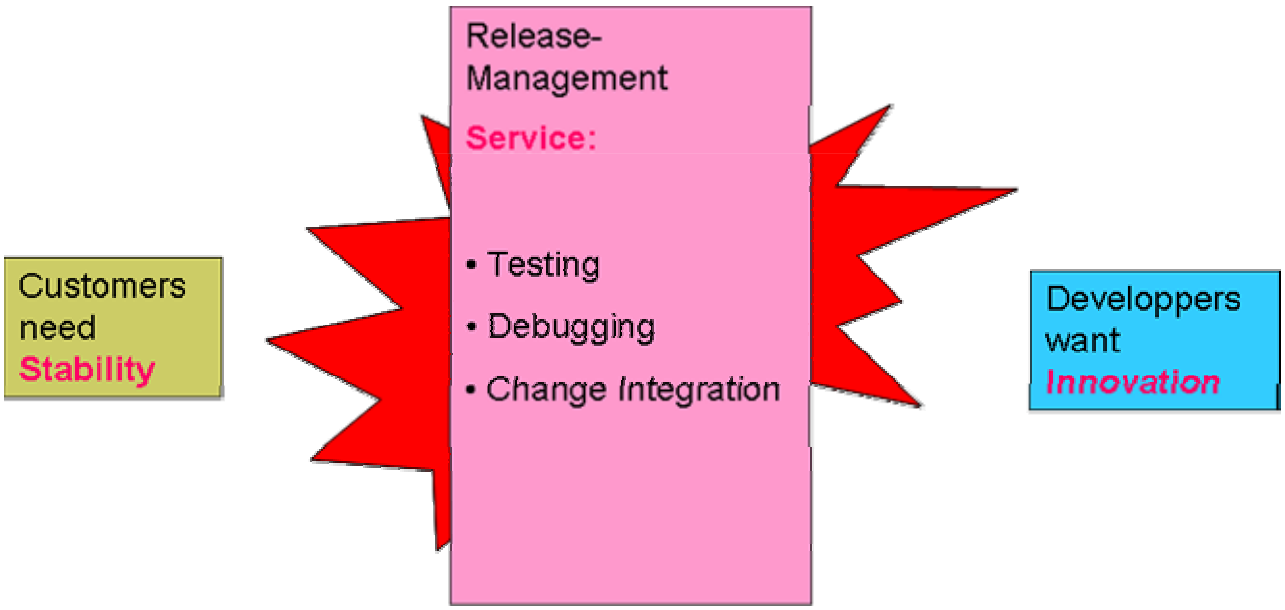

Figure 5: Release-Management

Before describing how Gump is used, we will give a brief overview of the tool.

Terms:

A **project** contains all the software, the documentation and files, which are needed to build, to test and to describe a closed portion of software.

A **repository** is a software pool on a fileserver, which is managed by a Version Control System like CVS or Subversion.

Continuous Integration Services take care of **dependencies**. One project can depend on other projects.

Gump needs a description of the repository and of all dependencies for each project that has to be taken into account in a run. The collection of all those descriptions, which are required for a run, is called a Gump **workspace**.

Along with dynamically downloaded software projects Gump manages a pool of static **packages**. i.e. pre-build software, which is either postulated not to change, or available only with a special license contract with the owner of the code, or otherwise not available or too difficult to download from a version controlled repository. One example of such a package is JAXP, the Java API for XML Processing.

Functionality:

The work within each run of Gump is done in four **phases** (like other CIT's):

1. resolve software dependencies
2. download projects from their repositories
3. build and test of projects
4. presentation of results

At the start of each build the dependencies of all projects in the workspace are resolved, which leads to a serialization of the builds. In the build phase the description of dependencies in the Gump workspace overrules the formulation of the classpath in the build-files and in the OS-environment. A reproducible dependency chain is assured, where the version of a package used in the run is well defined.

While the download of a project takes place, any files left over from previous runs are deleted. So each build takes place in a clean surrounding. If the actual download from the repository fails, Gump tries to continue the run with the files from the last successful download.

After all software of a project is created by the download, the builds and tests of each project are performed in the sequence of the dependency analysis. If a build is successful, the target of the build is stored into a result-directory. If the build of any required project fails, the build of all successor projects is cancelled.

After a run the results are analyzed and stored into a PostgreSQL-based database. The results become visible on a Web-Server as well. Gump offers the possibility to notify external developers via email if a new error condition is detected with their project.

Extensions to Gump

In the context of the release-management for grid-middleware, tools were developed to add and remove workspaces to and from Gump and adapt the presentation on the Web-Server. Now it is quite simple to check the software under different conditions, like different JDK-Versions, a different pool of static packages, etc. To do these tests, just a new workspace has to be defined.

A tool to extract a failed project and its prerequisites for further investigation has been developed.

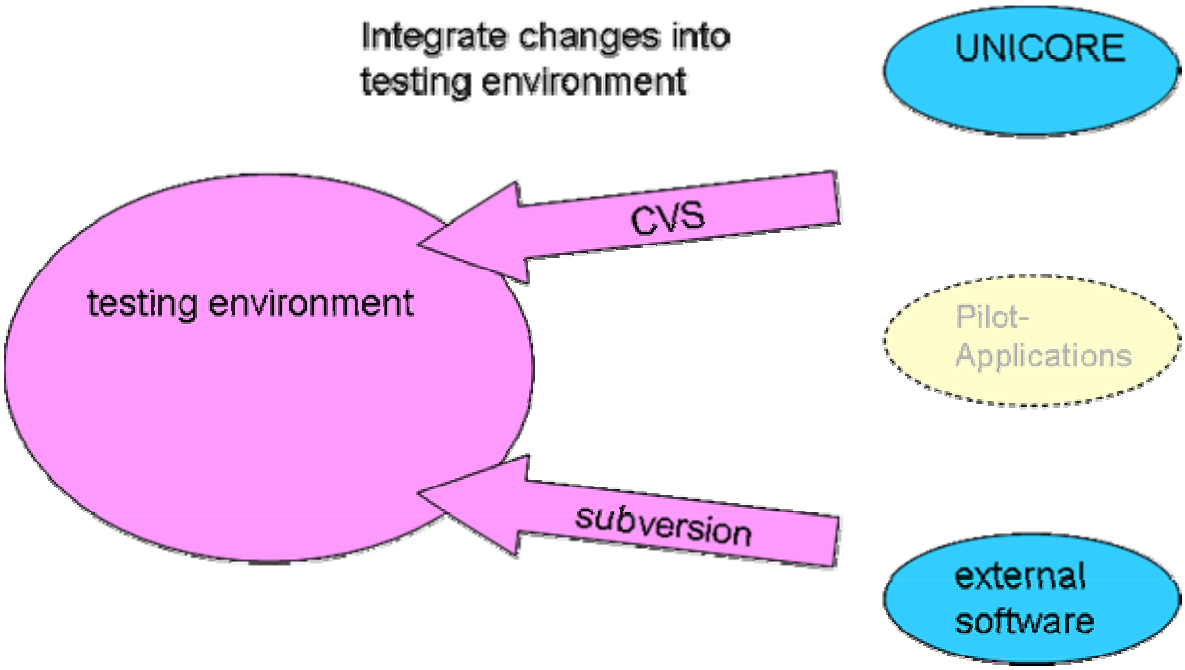

Figure 6: Continous Version Integration

2.2 A Grid based Service-Model for the IT-departments of SMEs

To open new horizons for research and development, it is necessary to provide ITC services in a very flexible and dynamic form with the highest reliability. Components from different sources have to be combined into complex solutions in an arbitrary way and without additional effort.

This flexibility in the dynamic provisioning of compatible services is only possible through the deployment of autonomous and reconfigurable systems that allow dynamic allocation and release of resources. Conformity with any required quality parameters (i.e. Service Levels) can be guaranteed by an automatic change or exchange of components. In complex systems it is not a good solution just to report the achieved quality-parameters and then to undergo an improvement process. Instead, an autonomous system must guarantee to work within the limits of the underlying SLAs. In the case of a breach of SLAs the respective components have to be replaced.

This vision of a service-grid however would not be complete, if we restrict it to functional aspects alone. Real progress will be measured in relation to the efficient and effective use of resources. It is important to keep competition between providers alive - particularly when services are provided by external partners. For this purpose, autonomous systems need a broker component that optimises the qualitative and economic parameters and autonomously selects the optimal provider. This aspect of shared use of resources raises additional questions.

In an environment as described, ITC-solutions are no longer provided by the acquisition of components or services but provided externally through the network. The complete business-workflow will typically span several organisations, at least those of the involved service-providers and customers. In such virtual organisations we will usually not only find friends, but also encounter competitors, and therefore it is essential that the interfaces between the components reflect the state of the art security-standards, especially concerning confidentiality.

On this basis, the discussion of advantages and disadvantages of classical outsourcing have to be posed in a different way. For basic services the discussion even becomes obsolete. The main question in future however will be, whether the integration and management of business-processes

will be done internally or outsourced.

A service grid combines the techniques of resource-grids with the concept of user-oriented services. A service typically consists of numerous components, each of which in principle can be provided by a different resource-provider as a utility. In a service-grid the resource-providers are not exposed to the grid-users directly, in principle he or she even doesn't have to know who they are. The grid-service provider thus is a generalized form of the grid-provider and he is responsible for the complete user-service, selects resource-providers and does the accounting and billing of resource providers and end users. Furthermore the service-provider is responsible for the choreography of the business-service.

The development of the provision of compute power, data and applications has made good progress during the last couple of years but the networking components are still a problematic area.

2.2.1 Shared Services

The main aspect of service-grids is the shared use of resources by several organisations and services. Whereas today service-providers and users still form a compact unit in many areas (e.g. HPC), in future these ties will be more and more loosened. No longer will the ownership of a resource be the determining factor for the capability to solve a certain problem – it will be the budget to access the required resources on the world-market.

A service-grid environment will put focus on the end-user rather than on the infrastructure.

2.2.2 Service-Model for SMEs

The IT-departments of small and medium enterprises have the problem that they are no longer able to produce all the services that their users request. The integration of services from external providers was rather difficult in the past. Service-Grids offer the concept of a seamless integration based on standards. From the perspective of a SMEs IT department, the service-scenario would look like Figure 16:

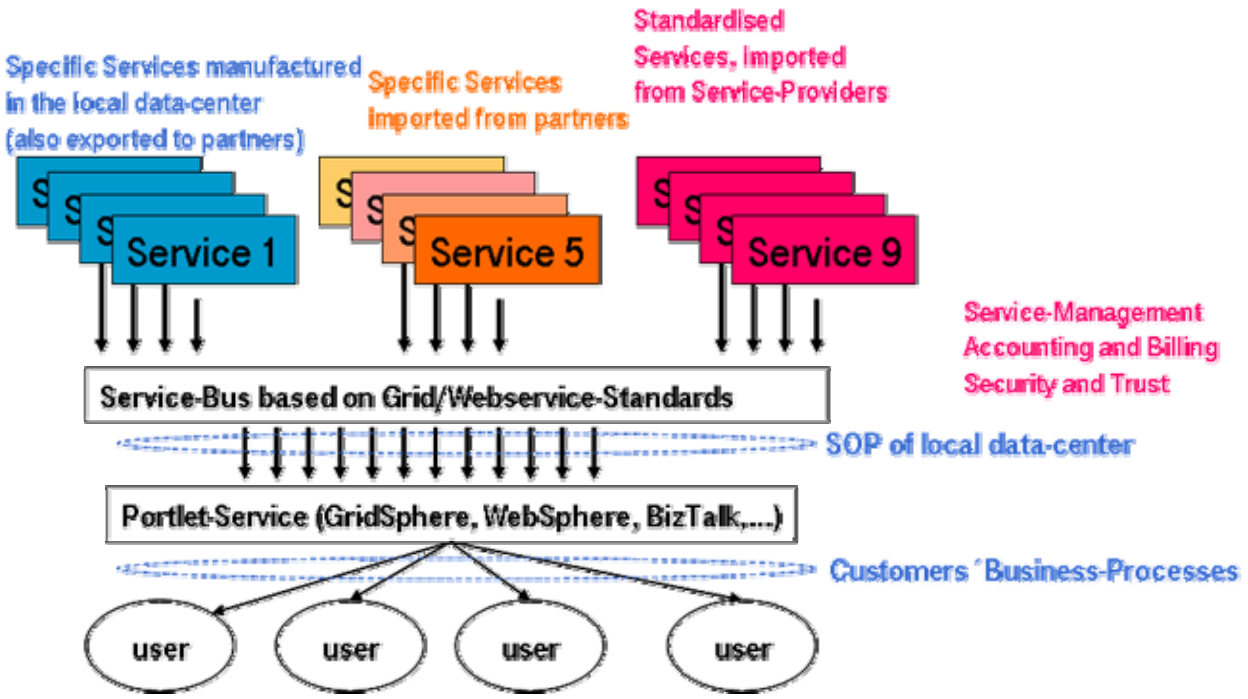

Figure 16: Service-Models for small and medium size IT-departments

2.2.3 Security and Trust

This proposed working-style for service-grids of course raises security issues related to functional safety, authentication and authorisation security and even more confidentiality. As the end-user very likely is no longer aware of with whom he is sharing resources, he has to rely on secured interfaces and methods. Therefore the standards have to be open and controllable.

Service-grids are not deterministic about the use of specific resources. In many sensitive areas (defense, medical) company-policies or legal regulations require that the complete processing-chain can be tracked and audited at any time.
Dynamic choreography also requires dynamic trust delegation schemes. A critical question here is whether only an explicit trust delegation is acceptable or also a delegation on the base of proxies and trust-metrics.

Even simple tasks, like where to save and backup data, are not trivial in a service-grid. The data have to be saved at a place that is accessible by the user in an easy and efficient way.

2.2.4 Accounting-Models and Globalisation

For the proper accounting and billing, each resource in a service-grid has to be underpinned with a price-model. This model can be dynamic in the sense that prices vary with system load or other parameters.

2.2.5 Cost-Reduction and Structural Changes

Unitl now the offshoring of IT-services mainly concerned programming and other development tasks. Complete services, due to their person- and process-centric nature were less affected. In future the offshoring of service-components is no more a general problem. Only the choreography of complete services and the mapping of complete business processes must happen in close proximity to the user (due to the high level of interaction).

In fact we face a similar development in IT-services as in the manufacturing industries previously. Processes, products and solutions are modularized in such a way, that the resulting standard-components can be easily exchanged. In our case the exchange can even be automated. The long term contractual relationships that are typical for ICT-services today in future will only make sense for the integration of modules to workflows. Here however the provider has to be integrated even more into the processes of customers than in the past.

2.3 Application-Service Providing: Past, Presence and Future

In this section we will discuss the deployment of Application Grids for scenarios where there are existing applications and workflows. Typically the user of an application or service belongs to a different organisation than that of the provider of a service or application. Whenever a partial or complete business-process is outsourced and the business process is realized in a service-oriented architecture, we speak about an application grid. The grid technology is used for linking the service consumer and service provider in a dynamic way. The SOA-concept allows the seamless integration of application-services provided by third parties into the service landscape of an organisation. For organisations and individuals that do not have their services integrated into a SOA, the classical portal concept can be used for the provision of the services.

Application Service Providing (ASP) was introduced about 6 years ago based on the technology of the WWW and first rudimentary portions of grid-technology. All the large service-companies and several research HPC-centres started ASP-projects of different flavours. Although successful from a technical point of view, the business success was very limited. In the hope that things can change and to preserve the investments made, the service companies continue to offer ASP, however they actually do not promote this model aggressively. Small startup companies, that started a business relying on ASP failed to create a successful business and have completely disappeared from the market.

2.3.1 Experiences with ASP in Technical Computing

In this section, the experiences, mainly of T-Systems, in the area of providing technical computing services based on the ASP model In the years 1998-2004 are described.

2.3.1.1 The hpcPortal

T-Systems was one of the early adopters of ASP in the area of technical computing. The so called hpcPortal has been operational for several years. Initially it shared a significant part of technology (security, file-transfer) with early versions of UNICORE.

What is the hpcPortal?

- hpcPortal is Application Service Providing (ASP) for CAE applications and License Service Providing (LSP).
- hpcPortal provides high-performance computational capabilities and application software via internet.
- hpcPortal provides software licenses via internet.
- All resources, e.g. use of CPU, memory capacity and software licenses, are charged according to actual use.

The hpcPortal was designed as the way to access supercomputers for customers with occasional needs, specifically:

- Need of high-performance computational capabilities for time-critical projects or very big models
- Computational capacity constraints
- The obligation to use a special hardware platform
- The temporary need of a software license

The following services can be accessed through the hpcPortal:

- Computational capabilities of supercomputers from all relevant manufactorers: NEC, IBM, CRAY, Fujitsu Siemens, Hitachi, HP
- Licenses from the leading CAE software companies
  ABAQUS, ANSYS, LS-DYNA, PAM-CRASH, PERMAS, MEDINA, FLOTHERM
- License services for Flotherm and Flovent
- Installation of the client‘s software
- Services like telephone support, online accounting

The typical way for new users to access hpcPortal has four steps:

- Link to hpcPortal (http://www.hpcportal.de)
- Register
- Call for the certificate
- Access / filetransfer / start of simulation / results

hpcPortal Security

- Authentification with digital certificates according to X509
- Coding of all customer data by SSL (Secure Socket Layer) or SSH (Secure Shell)
- Certification of the architecture and configuration of hpcPortal by the security experts of Inside Security Co.

2.3.1.2 License Service Providing (LSP)

License Service Providing allows users to get access to software-licenses that will be used on the user's own resources or on resources of a provider that is different from the provider of the software-licenses. This business model is not in contradiction to Application Service Providing where the application runs on the license-provider's resources, but complements ASP for applications or application components that have a graphics part like preprocessors etc. or require very specific hardware. Generally speaking, it gives the user the full flexibility in the selection of software-licenses and compute-platforms.

Today, the business model of almost all software vendors is based on the fact, that the customers pay for one or many licenses for a year, but the licenses are only used a small part of the time. They like to link their license to one node, one computer or best, one person. People at different places or in different business units of the same company should buy their own license and not use one grid license. FlexLM by Macrovision, which is the quasi standard for licensing engineering software just supports this node lock view and is not compatible with a grid infrastructure. Technically it would be rather simple to switch to another license model or even support different models at the same time. In order to bring an engineering software vendor to change his license policy, a change in the business model is necessary. The grid must not only be more useful for the customer, also a new profitable business model must be established which convinces also the software vendors.

There are several engineering applications within InGrid: CASTS, FEFLOW, FENFLOSS and INDEED. CASTS and FEFLOW are protected with the FlexLM license management system from Macrovision. INDEED uses its own license manager. FENFLOSS currently does not use any license management. The engineers of InGrid identified several critical issues currently hindering the adoption of Grid technologies in SMEs:

- **License Server Wrappers:** The strong market position and customer acceptance of FlexLM makes it imperative to support existing license models and licensing software in the D-Grid infrastructure. However, the closed source and closed APIs make FlexLM an unsuitable platform for innovation in licensing Grid software and services. As a result, a flexible framework is needed in which both existing licensing solutions and newly developed Grid specific license servers can be wrapped and services can be seamlessly integrated into the D-Grid. Figure 1 shows the design of the Service Licensing System. Based on FlexLM, a prototypical server wrapper has been created and tested with the CASTS application.

- **License Aware Service Wrappers for CASTS, FEFLOW, FENFLOSS and INDEED:** Prototypical license management extension to the service wrappers created for CASTS showed that the complexity quickly grows beyond a manageable threshold if services are to be used in a dynamic Grid environment. Hence, tool support is required which will allow non-Grid experts to integrate license management into their services. The license management support will be coordinated with the license support in both the service generator and workflow generator developed in AP3.2.
- **Grid License Models:** The current license models utilized by InGrid engineers are geared towards a small number of customers who utilize their software on a small number of dedicated machines on a regular basis. A license model allowing different forms of pay-per-use access to engineering services hosted in a generic Grid environment is required. This will enable the engineers to extend their customer base to infrequent users and at the same time reduce the in-house resource and management cost through outsourcing of the actual services to a service provider.
- **Grid License Server:** First experiments have shown that standard license management systems do not cope well with the highly dynamic Grid environment. Firewall and tunnel configuration and restrictive software locking make migration of licensed software unfeasible. Furthermore, the closed source and closed API nature of current licensing systems does not allow the integration of the required new Grid license models. A prototypical license manager will be used to test both the new license models as well as the integration of different license systems into the same Grid environment.
- **Information Services Integration:** The integration of different license management systems requires a uniform information dispersal mechanism to allow Grid users to transparently access services from different systems. A prototypical implementation of an MDS extension for FlexLM has shown that current Grid information services are well equipped to deal with this kind of information. The standardisation of the information presented is more of an issue. The integration of license information from a prototypical license manager with FlexLM will shed further light on this subject.
- **Scheduling Extensions:** Currently, Grid schedulers are not license-aware beyond the capability to start a certain job on a restricted number of nodes. The requirements posed by the engineers demand a more flexible scheduling system allowing for the on-demand acquisition of both resources and licences on a pay per use basis. Based on the Sun Grid Engine, a scheduling extension is planed which will interface with the MDS license information to dynamically acquire new licences to fit the number and location of rented nodes.
- **Accounting and Billing of Licensed Services**: Within the context of InGrid, the specific problems of license-accounting in grid-environments were analysed. On the base of this analysis a pragmatic approach for a solution was developped, that does not require any changes to the licensed software:
    - Setup of a firewall around the license-server.
    - The filtering-lists of the firewall, together with the logfiles of the license-manager deliver enough information to perform a simple accounting of license-use.
    - The solution is however not able to perform a selective accounting on the level of software-modules.
    - The solution is scalable in the same way as the license-management system itself.

  The described solution has been implemented and is currently in pilot-use in a production-environment.
- **Software Protection:** Unlike in the area of personal software licensing, the protection of the service side license verification code is not a high priority for the engineers in the D-Grid. The service providers like HLRS and T-Systems are considered trustworthy and the reverse engineering of application software to remove license management software is not a threat.

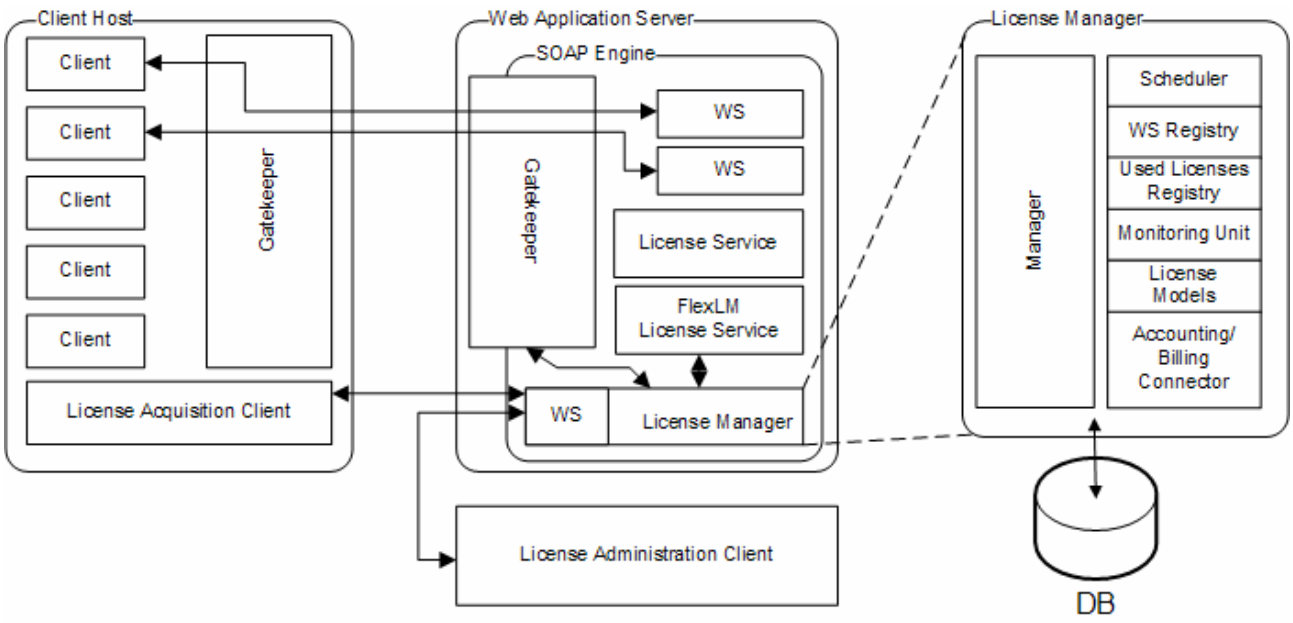

Figure 1: Service Licensing System

2.3.1.3 Lessons Learned

In the classical ASP model, the provider needs to negotiate contracts directly with Independent Software Vendors (ISVs) that give permission to offer the license for the ISV's software in an ASP-Model. This helps to keep the number of contracts with a customer that a customer must negotiate to a minimum as there is no direct contract between the ISV and the Customer.

The contents of the contract between the ASP and ISV include:

- Explanation of the HPCPortal
- Terms of Use
- Business-Issues (Sales & Marketing)
- Liability-Issues
- Support and Maintenance
- Duration, Cost, Confidentiality etc.

However, this leads to a very complex contract structure which is not commonly accepted by major U.S. Companies which insist on standard end-user contracts. European operations often are only authorised to offer standard end-user contracts need to refer to the main operation for new contract types. Sometimes ISVs may have contracts for ASPs that exclude other ASP providers or they only accept ASP models for fixed-term, named projects.

In each ISV there are promoters and enemies of ASP solutions. Usually the promoters are the challengers and have difficulties to succeed. The low revenues that are currently obtained from ASP models create a chicken-egg problem.

ASP contracts can have new advantages for ISVs, such as:

- ASP use can be an appetizer for a future "upgrade" to a standard license
- broadening of the ISV's customer-base to include SMEs who otherwise could not afford the software

- Flexible Overflow

2.3.2 Gap-Analysis

In this section the major problems of ASP as an early form of Application-Grids are identified and analysed.

2.3.2.1 Networking

Although the value for money ratio in networking actually improves much faster than that of processing-power, the inability to access appropriate network-connections has been an important showstopper for the industrial use of ASP-services in the past. Although industrial companies have no problem investing in networking capacity and bandwidth once they see a benefit for their business, there is still a critical situation due to the fact, that the provisioning model of networks is usually different between industry and academic institutions. Whereas research uses shared networks provided by the NRENs, industry uses dedicated connections directly from network providers. Naturally the latter is much more expensive and sometimes out of reach of SMEs. Network providers have recently begun to offer shared networks (that don't have very strict SLAs) to non academic institutions. What would help is the bundling of networking-capacity that is used for telecommunications during office-hours for fast data-transfers at night-time [7].

On the other hand the data-volumes are growing about the speed of processing-power and especially in time-dependent applications there is not much potential for data reduction.

What is missing for dynamic provisioning of networking-capacities is a scalable technology that can build fast networks out of affordable technology (e.g. DSL) i.e. a "PC-cluster" technology for networks. We actually do not have a networking-technology available that is affordable and scales over several orders of magnitude. Furthermore, the business-model of most network-providers is based on charging for the access-bandwidth and not for the use. Thus we still have a lot of room for improvement until networks can be provided as a utility in grid environments.

As software-tools like Gridftp easily could stripe one data-stream across several physical network connections, the real problem principally would reduce to a routing problem. However Gridftp was never designed to be used in sensitive environments and is not QoS aware at all. So what is urgently needed, is a multi-stream data-transfer facility, that from its implementation is state of the art concerning the security architecture and is able to interact with Quality of Service schemes. Allowing to the user-organisation to give priority to business-critical applications and thus to avoid conflicts between bursts and real-time applications as they arise today if Gridftp is used.

To close the existing gap on the lower layers, the involvement of companies like Cisco or Alcatel would be needed to develop the required technology components.

The technology would have to be underpinned with appropriate business-models, this is a topic for telecommunication companies.

2.3.2.2 Issues with Independent Software Vendors (ISVs)

ISVs are unsure of the effects of widespread ASP on their business models. Clearly, they will no longer be able to sell (lucrative) licenses to all the customers of an ASP provider, so they might lose considerable revenue. Whether this is offset by growing the user base (and the usage), is not clear. Therefore an economical model for license-management and pricing has to be developed based on

- Strategic Alliance with ISVs
- Collaboration with engineering consultants and research-institutions

- Development of user-community

ISVs are also critically dependant on a close relationship with their customers to guide the further development of the often highly specialized codes. This includes a significant amount of consulting-business that is typically performed by ISVs. The ISVs are also concerned about losing contact to their end-customers in an ASP model. It is therefore necessary to find ways how to get ASP going, but still have the ISV be in the contractual relationship with the customer.

A new model for ASP that takes care of these issues is therefore proposed in section 2.3.3.

2.3.2.3 Portals

The mid-term goal in most companies is to achieve fully integrated business workflows that are technically realized on the base of a service-oriented infrastructure. However, the situation today is, that in many companies it is still the state of the art to access applications through portal-solutions. These portals hide the technical infrastructure from the user. He only sees his application and can concentrate on his core-tasks.

Such portal-solutions typically interface to the application on one side, to the batch-System on the other side.

Prominent species in this arena are EngineFrame from NICE and Synfiniway from Fujitsu Systems.

These portal-solutions can only be used in Intranets, as they have no functionality to cross organisational boundaries or to deal with firewalls. Also they are often linked to a specific batch-system like to Platform´s LSF in the case of EnginFrame.

The restrictions mentioned above are overcome by some promising portal or portal-like solutions that have recently come up within some academic research projects and that is why the InGrid project should track their evolution and consider if they might be useful for the project. As examples we mention the RealityGrid project's Application Hosting Environment (AHE)[15], the Grid Execution Management for Legacy Code Architecture (GEMLCA), which has been developed within the CoreGRID project[16], and the HPC-Europa portal[17]. These interface solutions all have in common that they allow crossing organisational boundaries since for this purpose they use the functionalities of proven Grid middleware such as Unicore or Globus. Regardless of their conceptual and architectural differences, each of them strives to use standards wherever possible.

Only to give a few examples:

- The portlet technology [18] is used both by the workflow-oriented P-GRADE portal, which is based on GEMLCA, as well as by the HPC-Europa portal. Both approaches use the freely available GridSphere portlet container[19].
- The AHE server provides application-specific web services, which are consistent with the WSRF specification[20].
- Both the AHE and the HPC-Europa portal use the Job Specification Description Language (JSDL)[21].

Each of the given examples focuses on the integration of commercial or legacy applications whose source code cannot be modified for this purpose. For example, as to GEMLCA, server administrators deploy legacy applications as Grid services, which requires that the application parameters and environment values are described in an XML-based file. The HPC-Europa portal developers have demonstrated the automatic generation of an application-specific portlet from a simple XML description to start a simulation using the CFD software FLUENT.

It is a special feature of the GEMLCA P-GRADE portal that it supports workflow between different applications. While this makes this solution interesting for the user community targeted by InGrid, the HPC-Europa has the benefit that it has proven its concept of providing a single point of access with a unique graphical user interface for submitting jobs to heterogeneous computing centres to work successfully, even if the centres are using not the same middleware in the backend tier. For

this purpose a common job submission and monitoring interface had been defined, which has to be implemented by middleware-specific plug-ins. Plug-ins are already available for some Globus-based middleware systems as well as for UNICORE and GRIA. Although the HPC-Europa portal originally has been devised as a dedicated solution for providing a single point of access to the computing centres involved in the project, it generally appears to be interesting for heterogeneous environments where different resource providers have different Grid middleware deployed. Due to the modularity of the architecture, it is easy to add new resource providers or integrate other middleware software by just implementing the job submission and monitoring plug-ins. With respect to accounting it should be mentioned that a prototypic web-service based accounting service has been developed, which is also based on a common interface. This interface has to be implemented by a site that wants accounting information to be displayed to portal users, and there is no need to modify the accounting system already in production. It should be noted that besides the basic services (like single-sign-on, job submission and monitoring) other functionalities such as application integration and accounting are in a more prototypic state, but the development will probably be continued within other European research projects such as OMII-Europe 0.

2.3.2.4 Simulation in Distributed Environments

There is a lot of experience related to technical simulations in distributed environments in universities and research institutes. Industrial environments however are distinct in two aspects:

1. The lack of generally available high-speed networks at no additional cost. In industry such networks are only available if there is a proven business-benefit.
2. The formation of virtual organisations with a lot of security requirements. Business-relationships are much more dynamic than research-cooperations in academia. Also there is a much higher tension between cooperation and competition.

It is therefore necessary to take the lessons learned in the academic world and to translate them into concepts applicable for industrial user-communities. These concepts however have to start from a business perspective. This means that scenarios have to be identified which can benefit from the experiences made in the academic world in the sense that the underlying technology leads to

- Reduced Cost
- Better business-performance at the same cost
- New Business

These concepts and scenarios are then a base for initial consulting-actions.

As an example the requirements of an authentic scenario for grid-based process optimization in the casting sector are given:

An optimization of a casting process by simulation could involve up to five different players (individuals, institutions or working groups), possibly working from various locations and using different kinds of hardware/software platforms:

1. The casting engineer interested in producing a component of high casting quality. This person must have access to the preprocessor and postprocessor of the casting simulation program.
2. The quality engineer with expertise in the field of material properties, with measuring technology at his disposal for testing the quality of castings. This person needs access to the simulation results and thus requires access to the postprocessor.
3. The calculation engineer who is very familiar with the simulation software and is able to verify that the simulation is correctly defined in terms of initial and boundary conditions. For this person, direct access to both preprocessor and postprocessor is essential.
4. The specialist in computer-supported optimization, who supports the optimization of process parameters with high-performance optimization algorithms. This person has no

need of detailed knowledge of the simulation software, but rather configures and runs the optimization software.
5. The institution providing the computational resources, such as networked workstations or dedicated cluster systems on which the simulation and optimization software is run. This can, of course, be ensured by access to a remote computer cluster.

The casting and quality engineers are part of the same company, but frequently belong to different teams and are not necessarily working at the same location. Also in the case of the two simulation experts, the knowledge required is very different and these specialists may come from different institutes. While the calculation expert needs to have very good knowledge of casting simulation software, this is not a requirement for the optimization expert. The optimization expert selects the optimization program algorithms and parameters so as to achieve a good solution with as few simulation operations as possible (3-5). The computational resource provider plays no direct role in the simulation application. Decisions about allocation of simulation runs to computational resources and who is to provide the hardware required remains fully independent from the task. The simulations can run, for example, remotely on a grid resource like networked workstations or cluster systems. Important is that, alongside computer performance and data security, a system for billing the actual usage of resources is also provided.

From this scenario the following requirements for a compute-grid to become an attractive option for the casting industry can be deduced:

1. Grid technology needs to be able to provide the computer power of high-end workstation clusters over a limited period of time. This computer power must be available on demand and without long waiting times.
2. Since the execution of casting simulations may take hours (or even days) on a high-end computer cluster, the user must be able to monitor intermediate results while the simulation is ongoing.
3. To prevent the general distribution of proprietary data, secure data transfer and remote handling is mandatory.
4. Alongside the computer power grid, the technology has to provide online support by a simulation expert. The company's casting engineer and a simulation expert can discuss the simulation set-up and the simulation results online. This way, errors and problems during the set-up of the simulation as well as incorrect interpretation of the results can be avoided.
5. The grid technology has to feature a secure billing system, dedicated to accounting the CPU time and consulting expended for the casting simulations.

2.3.3 A new Business Model for Application Grids

2.3.3.1 Goals

The main goal is to create a model for a flexible access to applications based on existing and emerging Web-Service and grid standards, that is of benefit for:

- The End-User
- The Provider of the Software
- The Provider of the Computational Resources
- The Provider of the network-connectivity

This service should be easy to integrate into corporate IT Service infrastructures (e.g. based on a service-oriented architecture (SOA)) or used standalone vie Web-portals or user-friendly clients.

A secondary goal is the creation of a new market for grid services.

2.3.3.2 Concept

The concept is based on experiences and developments from the ISP (Internet-Service-Provider) world.

The development there started in a similar way to the current state of Application Grids as early providers combined infrastructure with content. Historic systems like the minitel in France or BTX in Germany were based on such concepts. Over the past two decades and the advent of the WWW, the development clearly went in the direction of a separation of the content from the infrastructure. There are providers of infrastructure such as AOL and T-Online, however nearly everybody is a content-provider.

We try to transfer this model to ASP, where again we have the situation today, that infrastructure and content (applications) are in the same hands.

Generic Infrastructure provided by Service-Providers

- HPC platforms
- Security
- Access
- Accounting and Billing (multi-source)
- Application hosting (analogous to the role of ISPs in Web-Hosting)

Content is provided by ISVs and other Application Service Providers (also may be interesting for research institutions) in fields such as Engineering and Life Sciences. There is no interference with standard- license business as the ISVs provide the content (applications) directly.

This leads to a model of Application Service on Demand with potential customers being SMEs and overflow capacity and capability for large customers.

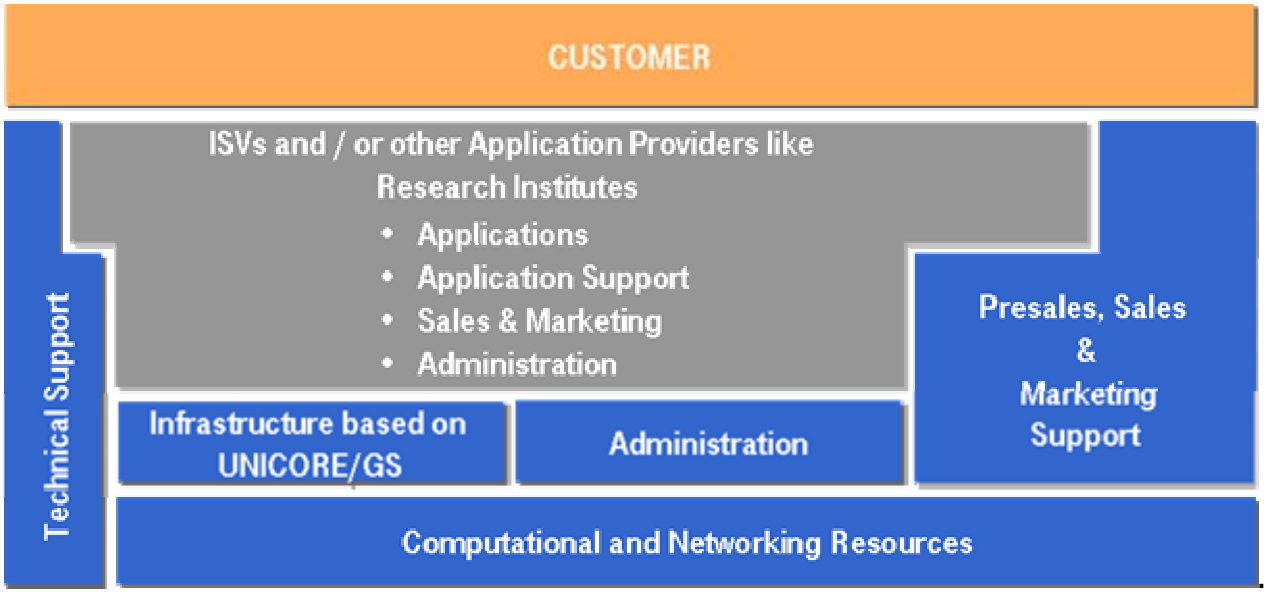

Figure 12: Second Generation ASP Business Model

Consequently the contractual relationships between the three parties are no longer a classical ASP chain (see figure 13)

Figure 13: **First generation ASP Relationship**

but now divides into 2 independent relations: to the ISV and to the provider of computing services (see Figure 13).

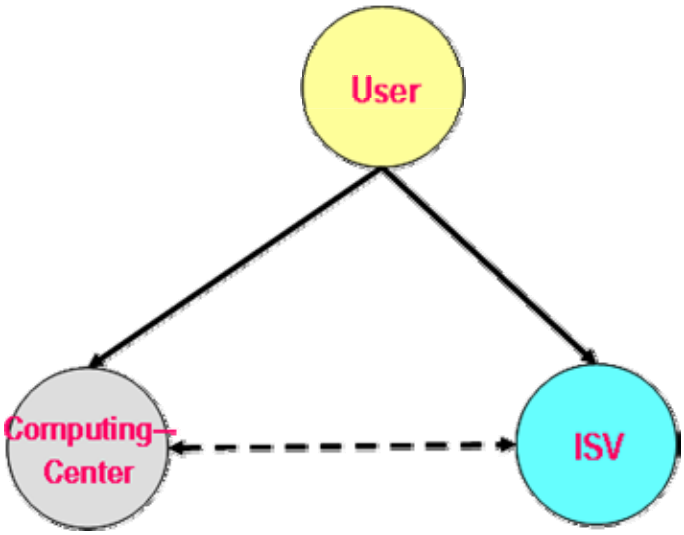

Figure 14: **Second Generation ASP Relationship**

There are still a number of problems to solve with this model. For instance, customers who use several software-packages will have to go out and talk with each of the ISVs.

On the other hand this model has the flexibility, not only to be well suited for established ISVs, but also to give startups, engineering consultants and research-organisations an easy market entry, as they can go to market without any financial or other investments in infrastructure and administrative tasks like accounting and billing. Therefore this model would definitively be a good tool for the knowledge-transfer from research to the industrial practice.

In general this model would foster the formation of partnerships to establish services and solutions for the market.

The service providers can offer additional infrastructure services like Information Lifecycle Management with the components Storage, Backup, Archive and Hierarchical Storage Management.

2.4 Service Level Agreements (SLA)-Management

In day-to-day business situations, consumer and service provider negotiate or agree on pre-formulated contracts in a written form in order to ensure that a certain quality of service is met by the provider, respectively that the customer does not assert unjust claims. Since this however is no *guarantee* for correct performance, the customer will generally want to compensate for possible losses from non-performance on service provider side by defining adequate penalties, i.e. penalties high enough to fully compensate financial losses of the customer - this, however, is an unrealistic assumption for real-life business scenarios.

Hence, in order to prevent such losses, customer and service provider want to monitor the progress of service execution so as to react accordingly to possible deviations from the original plan - this however is generally associated with a high work overhead on both service provider and customer side.

The concept of Service Level Agreements (SLAs) foresees the definition of electronic contracts that define specific quality of service (QoS) parameters that a service is to maintain during its interaction with a specific customer. Given the right system, SLAs would enable automatic observation of a specific service's performance and thus allow for timely reactions to critical deviations.

2.4.1 Technical and Business Requirements

There have been various approaches towards an electronic management of such contracts (e.g. [7]) to allow for autonomous observation whether the agreed upon terms are *maintained*. Most of these approaches, however, are either mere theoretical models without guidance towards realisation or completely dismiss important user requirements. It is hence relevant to first identify the main objectives forming the user requirements (cf. [8]):

2.4.1.1 Technical point of view

The main purpose of autonomous service level agreements consists in automatic enactment and observation of the QoS parameters, i.e. **monitoring** the performance of a service provider, and comparing it with previously agreed terms (**evaluation**). This information may then be used for fair **billing** of the service usage.

In addition to these obvious functionalities, SLAs may also be used as a criterion for the **identification** of services that fulfil a given set of (desired) QoS-descriptions. By publishing a set of potential SLAs that can be maintained by the service, a potential customer may identify whether the service meets his/her needs or not. Such information can also be used for an autonomous discovery process.

Since the concrete terms are generally up to **negotiation**, to e.g. agree upon the cost of service, such potential SLAs are just guidelines for the identification process. Negotiation, too, may be enacted autonomously, when the respective goals of the negotiation (e.g. "no cost higher than x") are defined. One of the main issues with respect to this consists in formulating the so-called "Business Level Objectives" representing these goals and policies in a manner comprehensible for the negotiation purposes.

2.4.1.2 Business point of view

An SLA management system may *under no circumstances* make data available that is protected by enterprise specific **privacy** issues, like e.g. the firewall settings of a specific computer within the network. Though this is generally a security issue that needs to be solved separately to the SLA management system, design of the latter should allow for the respective requirements and not make security realisation more difficult. In general, the system should not allow to access data that is not explicitly made available by the agreed SLA definition.

Similar to this, **internal policies** have to be respected by the system - besides for security issues like the ones above, this also includes statements about SLA-restricting behaviours, e.g. limitation of the performance so as not to overheat the system.

These issues are of particular relevance in connection with **self-management** of services: a service provider may want the system to automatically adjust the service behaviour (within a given limitation) in order to avoid SLA violations, thus always providing full QoS compliance - obviously, a SLA management system can and should not provide the means of autonomous self-management, but it can substantially support this task by providing the relevant information to a given administrative instance.

Since measured performance may have an influence on billing the service usage, customer *and* service provider would furthermore like to keep the system **neutral** so as to not result in biased evaluation of the data, as this would improve, respectively reduce the profit unfairly.

Finally, any business entity would want to maintain **privacy issues** with respect to the resources they expose to the Internet. This does not only concern general issues like hiding the infrastructure, respecting the internal policies etc., but may also involve SLA-related aspects since status information may be considered private by the service provider. Accordingly, any autonomous system has to allow for means of *neutralising* status information in a way that is still meaningful for supervision purposes, yet does not expose sensitive details.

2.4.1.3 Further requirements

Most approaches require the service provider to install or implement components on their system that allow for realising the respective functionalities - obviously, most companies or business entities will be reluctant to any changes to their systems which may cause worse performance and less compatibility with existing customers and/or business partners. Accordingly, an SLA management system should have as **little impact** on the existing infrastructure as possible.

Along the same line, implementation and maintenance of the system, including adaptations to different SLAs should be easy and straightforward, thus requiring as **little administrative overhead** as possible. Since SLAs base on a "generic" terminology that allows both providers and users to communicate on the same issues, an autonomous management system *should* provide the means for easy mapping of generic terms to infrastructure-specific ones.

2.4.1.4 SLA Management on Service Provider Side

If the user has not (and should not have) direct access to the actual system status of the provided service, the service provider *has to* extend his services so as to allow for status monitoring. Generally, the required tools are available for most users (like e.g. WMI [9], [10]) which comes with the latest Windows Versions or Ganglia [11] which is an open source tool for Linux). Since SLA specifications differ from the data provided by these tools, some "parsing" functionality needs be supported.

Note in particular that functionalities offered by service providers may involve more than one resource thus requiring more complex aggregation of status information, i.e. a combination of local monitoring and parsing instances may take over this task. This may *principally* be realised using a

proxy, in which case it gathers the required information and converts them according to the metrics provided by the service owner - please refer to [12] for more information.

Furthermore, "local" management can be extended to allow for "preventive" SLA management: by lowering the parameters agreed upon in the SLA, a service administrator or self-management system could be informed of an increased violation risk *in time*, thus being able to take corrective measures. Again, a service provider *may* use a proxy for this functionality by providing an additional, "preventive" SLA, this way saving the integration effort - generally, it may be assumed though, that a company would want to keep the respective process short and quick, without having to access external parties first.

2.4.1.5 Neutral Notary

A third party "notary" may store the SLA during the actual usage of the service by the customer for reference. This way it is ensured that potential differences in the two SLAs can be solved with the help of a neutral party that maintains a valid copy of the *original* SLA document. A nowadays more common approach consists in enacting a complicated **signing** protocol that leads to an encoded SLA that may be read, yet not altered. Here the Notary may act as a "witness" or actually perform the signing process based on keys provided by the respective parties.

Likewise, third party "logs" could keep an account of all the violations that have occurred during the actual operation and serve as a reference to e.g. support billing on the basis of service performance.

2.4.2 SLA Management Components

In the following we describe the main components of an SLA Management system. Service providers as well as service consumers have to manage their SLA relevant information before, during and after the execution of a service.

SLA Manager: The VO as well as the Service Provider uses an SLA Manager to configure and coordinate the components of the SLA Management sub-system. In particular the SLA Manager is responsible for the configuration of SLA Monitors and SLA Evaluators. The Service Provider’s SLA Manager may also be responsible for converting abstract into concrete monitoring terms.

SLA Template Repository: A service provider maintains SLA templates in an SLA Template Repository. SLA Templates describe ranges of QoS parameters that a service is in principle able to achieve. In this way an SLA Template is an extension of the service description and can be used in the service discovery process.

SLA Negotiator: Once suitable services have been found, the VO will initiate a negotiation process with the service provider. The SLA Negotiator supports the process of reaching an agreement by implementing a suitable protocol. The actual logic to decide when a negotiation is considered successful lies outside the SLA Management subsystem.

Notary Service: A Notary Service is a trusted third party component that witnesses the signing of SLAs between service providers and service consumers. The Notary stores the signed SLA for later reference in an SLA Repository.

SLA Signer: This local component implements one of the sides in the signing protocol chosen for the VO and admitted by the Notary.

SLA Monitor: We can distinguish service provider internal monitors and external monitors, which should have the status of trusted third-party services. Internal monitors have direct access to the applications and resources they monitor. External monitors on the other hand can only inspect web services at their interfaces and usually lie outside the control of the service owner. Both kinds should be able to aggregate the metrics produced by other monitors into composite (higher-level) metrics.

SLA Evaluator: An SLA Evaluator receives monitoring information and compares them to the promised values. In case of an underperformance as compared to the SLA parameters, the SLA Evaluator notifies the SLA Manager.

SLA Performance Log: This component accumulates data on the performance of SLA monitored services. The logged data is used for inspection and accounting purposes.

2.4.3 Different Levels of interaction in SLA Management

In the following sections, the overall interactions from service discovery up to billing shall be outlined so as to provide some insight into the processes involved.

Identification: A (potential) customer may query UDDI-like repositories with a description of the required service to identify potential service providers. If some link to the service provider's SLA template repository is retrieved with this information, the customer is able to request the SLA templates from the providers of interest. Such templates represent the quality that may be *principally* maintained by the service - on basis of these, a customer may identify the most suitable service for his/her purposes, i.e. the one which can fulfil the QoS-requirements best. It may occur, however, that *no* service meets all the requirements, in which case an adaptation of the constraints should take place.

Negotiation: In most cases it will be necessary to negotiate the actual SLA parameters, as the service provider will allow for some flexibility with respect to the quality and the obligations that the services may fulfil (e.g. lower price for less memory usage etc.) In addition to this, a template is no *guarantee* for a provider being able to maintain the described quality at the time of demand - accordingly, the negotiation process also serves the purposes of verifying availability.

Once an agreement is reached, the negotiated SLA will be stored in a Notary (see above) and/or signed so as to prevent further modifications. Theoretically the service is now ready to be used by the customer, though generally this is not a requirement. In some cases, it might be that the customer wants to use the service at a later point in time (e.g. a month, a day) which will obviously influence the negotiation process and entails that the service needs to be reserved.

Negotiation is generally difficult to automate as it requires a lot of intelligence to adapt the parameters in a way that represent the Service Provider's ''Business Level Objectives'' in an optimal fashion.

Configuration: In order to fulfil the agreed upon quality of service, the service provider configures his service accordingly - generally, such process is performed manually by the service owner, respectively administrator. Along with additional data about the infrastructure and how the configuration influences the quality of service etc., the configuration process may be considered a reverse of monitoring.

As mentioned before the customer would want to receive status reports in a certain time-interval. To this end, performance data needs to be gathered regularly from the service provider and compared to the agreed upon quality terms. Violations of these terms will lead to a notification being sent to all involved parties (customer and service provider) who may take according actions.

Accordingly, these internal components may take over the task of **business monitoring** too, where breaches of the original SLA are to be identified. However, the customer may not *trust* the service provider to evaluate the data correctly (''neutrality''), whilst the latter will not want to expose (sensible) system data to the customer (''privacy'').

Enactment: Status information, as provided by any data provisioning unit, is generally not identical to the parameters as defined in the SLA document, some form of conversion and in most cases even aggregation is required to make the data understandable for evaluation purposes: most service providers will offer functionalities combined of various resources - in the simplest case something like an analyst and his/her computation resources - which status' information is meaningless regarded separately. Since such conversion is directly dependent on the data

provider and since the service provider may want to "neutralise" sensitive information, it is logical to assume that the respective parsing capability is supported at service provider's side.

Violation handling: In order to prevent the actual contract breach, the service provider will want to re-configure his system so as to lower the respective risks. We hereby assume that some form of intelligent "Execution Management Service" may achieve this, by e.g. moving the process to a different host or freeing resources.

Even though we recommend to pursue TrustCoM's approach in separating SLA and consequences ("policies"), during negotiation all terms and conditions need to be approved so as to form a meaningful (and binding) contract. In the case of actual business violations, different reactions are thinkable and need be agreed upon during contract negotiation.

Sometimes the service provider may request verification of a violation. For example, if the evaluator detects an violation but the service provider itself has not observed one. In such a case, the above mentioned Notary may serve as a "neutral" party to solve issues by verifying the respective SLA documents.

2.4.4 Future topics

An SLA Management System that guarantees a clear distinction between service provider, customer and the actual SLA Management functionality, allows for easy modular "plug'n'play" usage, and hence observes the "little impact" objective. Since common system monitoring tools ("data providers") like WMI and Ganglia are easily available, configuration of the management system may be realised by simple deployment of the SLA document, which keeps maintenance overhead low.

However, the disadvantage of any SLA management system can not be completely overcome: the service provider needs to configure his/her system so as to allow monitoring from the outside. Though e.g. WMI allows for remote access to system status information, no company would want to allow such access. Accordingly, an interface needs to be supplied that provides the accessible information and hides all other data, thus enacting internal policies and privacy issues. Such an interface, of course, allows for manipulation of the data again and hence poses a similar threat with respect to confidentiality of the data, if no respective precautions are taken.

First implementation results are available from TrustCoM and NextGrid and have provided valuable insight into further implementation details - in particular with respect to the profiles of the underlying SLA specification [13] and its business requirements. It is still outstanding work to examine widely acceptable SLA parameters to reach something like a global understanding of SLA descriptions. Semantic web technologies (ontologies, dynamic logical derivations etc.) should be exploited in the long run to provide more flexibility in the definition of terms.

Other issues to solve include

- the calculation of dependencies between different SLAs, like e.g. when a company offers services that themselves are composed of services provided by other entities (subcontracting) and how these contribute to the overall SLA fulfilment.
- the possible ``richness" of SLA templates required for identifying services, respectively the enhancement thereof will have to be examined carefully so as to allow for more information to be provided.
- the legal aspect of SLAs with respect to an overall contractual framework need to be explored to make SLAs legally binding in real eBusiness scenarios [14].

Current IST research projects are looking into these issues, trying to come up with solutions that cover additional criteria, like security-related factors, scalability etc. This will be observed and adopted to the InGrid project.

2.5 Information Lifecycle Management

In an industrial commercial environment requirements are different from research activities which may be fulfilled on "best effort basis".

Some major requirements need to be considered:

- Projects need to be concluded within a fixed time frame.
- Projects are targeted with respect to costs.
- Results need to be documented and archived for product lifetime.
- All data need to be treated as confidential.
- etc.

Those requirements call for agreement on service levels. Depending on the service level requirements, the cost will vary. Due to the fact that for some resources requirement for availability and performance are high and therefore expensive, other resources may have lower requirements allowing for lower prices. Depending upon the project requirements means need to be available to find the adequate balance between availability and performance vs. cost aspects.

2.5.2 Data Management Components

2.5.2.1 Data Repositories

Basic assumption is that all data relevant for the engineering community is placed in data repositories. Those data repositories are managed professionally according to well defined service levels. Some of these service level parameters are:

- Latency in data access
- Bandwidth for data access
- Data safety means
- System availability

Depending on service level and cost structure the price for data storage and access will vary.

2.5.2.2 Data Creator

Newly generated data shall be made commonly available to the community. The person inserting data into repositories need to be given the choice where to place its data according to service level requirements and cost constraints. Mission critical data may be replicated in different data repositories in order to comply with availability or access requirements.

The data creator is also required to provide all semantic metadata in order to access the proper data object later.

2.5.2.3 Data User

The data user will find the proper data objects according to the semantic information provided by the data creator. If a single data object is replicated on different repositories, the data user is to be given the choice, which copy to use. Criteria again are service level and cost for data access.

2.5.3 Brokerage

A data creator as well as the data user is faced with the task to find the most suitable resource for data storage. In a GRID environment with dynamically changing resources a brokerage component is required.

2.5.3.1 Insert Process

Upon request of a data creator to insert a data object into data repository the broker will provide a list of available resources. For each resource the service level and the corresponding price will be presented. It is assumed that the data creator will be charged for data storage by the agency operating the repository.

2.5.3.2 Retrieval Process

Data retrieval is based on the metadata provided by the creator. Running the metadata directory is considered to be allocated to the broker. Upon retrieval request, the broker will present a list of copies for a specific data object together with the service level and the pricing information to be paid by the data user.

2.5.4 Security

Within the context of this paper, a user is considered member of an organizational unit. All security efforts are based on organizational unit in order to comply with changing allocation of individuals to organizations or tasks and projects over time.

The creator (organization) is considered to be the owner of a data object. The owner is responsible for granting and/or removing access permission to a data object or a collection of objects. The administration of individuals and organizations as well as the access control is considered to be a brokerage task. Authentication is based on a single person, while this person is authorized to use all resources available to all organizations/projects to which this person is assigned. All user administration as well as access control is considered a brokerage function.

2.5.5 Accounting/ Billing

A commercial environment for data handling needs to charge costs to the organization which does request a service.

Data repository costs as well as brokerage fees as well as costs for data usage are charged:

- Costs for data insertion are charged to the organization inserting and owning data objects.
- The organization retrieving data will be charged with repository and brokerage costs. In addition
- it will be charged with usage costs to be paid to the data owner.

Due to the fact that the broker component is involved in all data handling processes, accounting and billing will be handled by the broker facility

3. Administrative Requirements

3.1 Accounting and Billing

Accounting and Billing may be topics of less importance during the externally funded phase of a project. When it comes to a sustained service however, business relationships have to be built up and accounting and billing are essential requirements for a commercially successful service and thus a base for sustainability. This is especially true in engineering, where typically we deal with commercial values.

In Grid-environments, there are some specific requirements, that accounting and billing has to fulfil to get acceptance in a business oriented community:

- Resources owned by different providers have to be accounted and billed in a consistent way
- Accountable services may span multiple resources and thus multiple organisations
- Resources and thus resource-providers can be exchanged in a very dynamic way
- The end-user expects one bill for all the delivered services and not separate bills from different resource-providers.
- Organisations can be resource-providers and end-users at the same time (n:n relationship)

3.1.1 Accounting

The requirements of grid-accounting were analyzed on this base and cross-checked against the capabilities of the accounting-systems that service-providers have developed for their classical service business. Besides the fact, that in grid-environments the dynamic behaviour is more automated than in classical environments, there are no principally new features required. Specifically the consistent accounting of distributed and heterogenous resources that are owned by different organisations is state of the art today and can be handled with existing tools.

The only exception is the accounting of software-licenses in Grid-environments. However this is part of the generally unsolved problem of grid license-management.

3.1.2 Billing

Billing of services for which other organisations have complete responsibility (and do not only contribute components) is unusual in IT-services today.

In other markets however, such a situation is common. Specifically the mobile telecommunication market has very sophisticated solutions for this purpose. Although there is a fierce competition in the mobile communication-market, the service-providers have managed to set up infrastructures and to define interfaces for a global exchange of accounting-data and cross-billing. In consequence, the end-user today gets one bill from his or her contract partner, independent of the organisations which delivered the services.
The investigations on the transferability of solutions from the mobile-communications market to the grid-services market are ongoing in InGrid. So far no new requirements have popped up that would justify the development of new solutions. A difficulty for a direct transfer however could be the sheer differences in volume of the coming up grid-market and the well established market for mobile communication.

3.2 Legal Problems

During the work performed in WP 4.2 so far two legal issues have been raised:

- Under what conditions are public research institutions allowed to offer services on the market?
- Can commercial service-providers offer services based on components or resources provided by public research organisations without conflicting with commercial laws?

3.3 Identity Management

As the formation of cooperations and business-relationships in engineering is very dynamic, centralized approaches as they are possible in relatively static communities are impractical. An integration of the meta-directory systems that most organisations have today with the Shibboleth service of DFN seems to be sufficient and appropriate to fulfil the requirements of the engineering community.

4. References

[1] UniGrids project, http://www.unigrids.org.org/

[2] NextGrid project, http://www.nextgrid.org

[3] "The Role of the Business Model. in Capturing Value from Innovation:. Evidence from Xerox Corporation's. Technology Spinoff Companies." http://www.cvn.columbia.edu/jl/readings/Chesbrough_ICC_2002.pdf

[4] "The utility business model and the future of computing services", M.A. Rappa, http://www.research.ibm.com/journal/sj/431/rappa.html

[5] http://www.quickmba.com/entre/business-model/

[6] "Seven open source business strategies for competitive advantage", J. Koenig, http://management.itmanagersjournal.com/print.pl?sid=04/05/10/2052216

[7] Asit Dan et al. Web Services on demand: WSLA-driven automated management. *IBM Systems Journal*, 43(1), 2004

[8] Anbazhagan Mani and Arun Nagarajan. Understanding quality of service for Web services. http://www-128.ibm.com/developerworks/webservices/library/ws-quality.html.

[9] Craig Tunstall and Gwyn Cole. *Developing WMI Solutions: A Guide to Windows Management Instrumentation*. Addison-Wesley, 2003

[10] Microsoft. Windows Management Instrumentation. http://msdn.microsoft.com/library/en-us/wmisdk/wmi/wmi start page.asp.

[11] Matthew Massie, Brent Chun, and David Culler. The ganglia distributed monitoringsystem: design, implementation, and experience. *Parallel Computing*, 30(7), July 2004

[12] J. Haller, L. Schubert, and S. Wesner. Private Business Infrastructures in a VO Environment. In *eChallenges 2006 proceedings, in print*, IOS Press, 2006

[13] Michael Wilson, Alvaro Arenas, and Lutz Schubert. The TrustCoM Framework for trust, security and contract management of web services and the Grid - V2. http://www.eu-trustcom.com/DownDocumentation.php?tipo=docu&id=242

[14] Tobias Mahler, Alvaro Arenas, and Lutz Schubert. Contractual frameworks for Enterprise Networks and Virtual Organisations in E-Learning. In *eChallenges 2006 proceedings, in print*, IOS Press, 2006

[15] The RealityGrid project, http://www.realitygrid.org/AHE/index.shtml

[16] The CoreGrid project, http://www.coregrid.net/

[17] The HPC-Europa project, http://www.hpc-europa.org

[18] JSR 168, http://jcp.org/en/jsr/detail?id=168

[19] GridSphere, http://www.gridsphere.org

[20] The WS-Resource Framework, http://www.globus.org/wsrf/

[21] Job Specification Description Language, https://forge.gridforum.org/projects/jsdl-wg/

[22] OMII-Europe, http://www.omii-europe.com