

Towards SLA Based Software License Management in Grid Computing

Jiadao Li

Jiadao.Li@scai.fraunhofer.de

ERCIM

2004 Route des Lucioles, BP 93 06902 Sofia Antipolis

Wolfgang Ziegler, Oliver Wäldrich

{Wolfgang.Ziegler, Oliver.Wäldrich}@scai.fraunhofer.de

Department of Bioinformatics

Fraunhofer Institute SCAI

Schloss Birlinghoven, 53754 Sankt Augustin

Daniel Mallmann

D.Mallmann@fz-juelich.de

Research Centre Jülich, Jülich Supercomputing Centre (JSC)

D-52425 Jülich, Germany



CoreGRID Technical Report
Number TR-0136

June 25, 2008

Institute on Resource Management and Scheduling

CoreGRID - Network of Excellence

URL: <http://www.coregrid.net>

Towards SLA Based Software License Management in Grid Computing

Jiadao Li

Jiadao.Li@scai.fraunhofer.de

ERCIM

2004 Route des Lucioles, BP 93 06902 Sofia Antipolis

Wolfgang Ziegler, Oliver Wäldrich

{Wolfgang.Ziegler, Oliver.Wäldrich}@scai.fraunhofer.de

Department of Bioinformatics

Fraunhofer Institute SCAI

Schloss Birlinghoven, 53754 Sankt Augustin

Daniel Mallmann

D.Mallmann@fz-juelich.de

Research Centre Jülich, Jülich Supercomputing Centre (JSC)

D-52425 Jülich, Germany

CoreGRID TR-0136

June 25, 2008

Abstract

Software protection and licensing are important topics for both the independent software vendors and software users. In Grid environments, the use of license protected applications is almost impossible and becomes a challenging task. The reasons are twofold: (i) there are no business models of the independent software vendors for the Grid and (ii) there is no licensing technology suitable for Grid environments. In this paper, the state of the art of license management in software industry as well as the current practice for managing the licenses in Grid computing is presented. The challenges and requirements of managing the software license in Grid environments are identified. Two general models developed in the European projects BEinGRID and SmartLM for managing the software licenses in Grid computing based on service level agreement (SLA) are presented.

1 Introduction

Grid computing is considered a cornerstone of next generation distributed computing, which is defined as “coordinated resource sharing and problem solving in dynamic, multi-institutional collaborations”. Current Grid computing infrastructure is built in accordance with the service oriented architecture (SOA [17]) paradigm. A service-oriented architecture is one in which all entities are services. The services [33] may include both traditional resources (e.g., compute services offered, network bandwidth, or space on a storage system) and virtualized services (e.g., database, data transfer, simulation, licenses), which may differ in the functions they provide to users but are consistent in the manner in which they can deliver those functions across the network. In Grid computing, the co-allocation of different kinds of services (compute resources, memory capacity, applications, etc) usually belonging to different resource/service providers is needed in order to satisfy the needs of a complex job. Virtual organizations (VOs) are dynamically created according to the requirements of different jobs (users). The jobs will potentially be scheduled and relocated across the whole Grid, and users might have no knowledge about the places where their jobs are to be executed. The current practice of managing and using the software licenses is limited to the local administrative domain while Grid infrastructure is usually stretching across domains as we introduced before. As a result, the users often can not use Grid resources because the applications they need are not licensed on the remote resources.

To this end, software licenses become a major obstacle for the users to use Grid infrastructures. Moreover, if possible at all, using Grid resources with the current licensing and pricing models of the independent software vendors (ISVs) is most often more expensive than running the application locally. To leverage use of Grid resources for license protected applications flexible pricing and licensing models for the benefit of both the software vendors and users are needed [8], [31].

To better integrate licenses into Grid infrastructure software licenses should become schedulable and manageable services as other services. Service level agreements (SLAs) have turned out to be a valuable instrument for agreements on the terms of service usage and reservation in different administrative domains. We see a SLA-based license management approach as the appropriate solution to the licensing problem in the Grid. Two different, complimentary approaches will be briefly presented, which are currently under development in the European projects BEinGRID and SmartLM.

The rest of the paper is organised as follows: In Section 2 we briefly give an overview on existing technologies. Section 3 presents the existing models for licensing and pricing. Current work on two different, complimentary approaches is presented in Section 4 and Section 5 concludes the paper with an outlook.

2 Related Work

To the best of our knowledge currently there are no other approaches to overcome the licence problem in the Grid, except for the approaches for license scheduling described in Section 4. However, software protection and license management is an important topic for the software industry and under continuous discussion, e.g. SoftSummit [19] is an executive conference dedicated to strategies and best practices for software licensing, pricing as well as application packaging and license tracking. Within a single administration domain, the centralized license management paradigm is suitable and efficient to manage the licenses. To this end, there are many license management systems from different companies available, e.g., LUM from IBM [6], License asset manager from TeamEDA [9], iFOR/LS from HP [28], RLM [14] from reprise software, Open iT [12], license tracker [20], Sentinel software protection and licensing package from SafeNet Inc [16], FLEXnet manager from Macrovision [24, 23, 26, 25, 27]. The license management system should be in charge of the whole life cycle of the license use, which includes: demonstration licenses, evaluation licenses, full-production licenses, product updates, maintenance releases, etc. Most often used common license models supported by these license management toolkits are described below. Some of these mechanisms need the license servers and vendor daemons, while other mechanisms do not need them.

To this end, FLEXnet is a typical license management toolkit used by many ISVs. The FLEXnet manager provides numerous kinds of license models supporting different scenarios. The ISVs decide and offer the different license policies based on the currently available models in FLEXnet in order to satisfy the needs of their users while increasing the revenue at the same time. Today, we see limited incremental progress towards more flexible licensing solutions on the side of the companies providing the licensing technology, and even less progress on the side of the ISVs. It must also be mentioned that the price for both licensing solutions leveraging flexibility of the end-user and pricing of the ISVs increases with degree of flexibility, which clearly is a show-stopper for usage of commercial applications in

Grids.

3 Current license and pricing models

In this section we give an overview on current models. We use licensing models (and terminology) of FLEXnet, as this is the technology for software licensing most often used by ISVs and the other licensing technology providers offer either subsets of models supported by FLEXnet or quite similar models.

3.1 Typical License Models

The basic license models provided by FLEXnet [24, 23, 26, 25, 27] presented next can be combined to create new license models.

- **Node-locked licenses:** Node-locking means the software can only be used on one machine or a set of machines. There are two types of node-locked licenses: uncounted and counted. For the counted node locked licenses, a license server and a vendor daemon are necessary.
- **Floating (concurrent) licenses:** Anyone on the network can use the licensed application, up to the limit specified in the license file (also referred to as concurrent usage or network licensing).
- **Mixed node-locked and floating licenses:** Uncounted node-locked and concurrent usage licenses can be mixed in the same license file, therefore more flexible usage models can be derived.
- **Demo licenses/evaluation licenses:** Properties of an evaluation license may include: (i) Limited product functionalities or features, (ii) Limited number of uses, (iii) Expiration date.
- **Usage-based licensing:** A quite important license strategy in which the actual usage patterns are monitored by the license management system, and billing or auditing are based on the actual usage data. FLEXnet Licensing supports several usage-based models, e.g: *Overdraft*: allowing the ISV to specify a number of additional licenses which customers are allowed to use in addition to the licenses purchased; *Pay-per-use*: allowing the customers to pay for the effective usage of the licenses, which can be audited based on time, the number of transactions, etc.
- **Mobile licensing:** Used when users want to run an application on a machine that does not have a *continuous* connection to a license server system. These situations can include:
 - Working on a laptop; or using a computer both at work and at home or off-site; or working from several different computers not connected to a license server system
 - Fulfilled from a prepaid license pool: The license is fulfilled from a prepaid number of license-days for the usage period.
 - Node-locked to a user name: If a license is to be used exclusively by one user on different machines, that license can be node-locked to the users user name.
 - License rehosting: if an end-user want to move a license without using one of the other mobile licensing methods. In this model, a new node-locked license certificate for each new machine should be generated.
 - Hard-mobile: Mobile license usage is controlled by a FLEXid. If the FLEXid is attached to a license server system, then the use floats on the network. To temporarily transfer the license, the user moves the FLEXid from the server to a standalone machine.
 - Soft-mobile: Licenses are temporarily transferred to a license server system on the mobile laptop. The FLEXenabled product uses an encrypted local file, placed there by the license server system, to do check-outs during the usage period.
 - License borrowing: A license can be borrowed from a license server system via a special checkout and used later to run an application on a computer that is no longer connected to the license server.

3.2 Current Business and Pricing Models

ISVs usually define different licensing policies with respect to their different products and target customers, e.g., some software vendors will provide the enterprise software, while others provide non-enterprise business software or consumer software. For some software products ISVs may expect a large number of customers, while the use of other products is limited to a specific user group. The open source software model also has great impact on the software licensing and pricing. To fully understand the models it is important to realise that the ISVs do not sell software but the right to use a certain software under dedicated conditions, which on the other hand implies that the customer only buys the right for limited usage of the software governed by the model he is willing to pay for. With the current move to multicore CPU technologies with tens or hundreds of cores in one CPU, we expect strong impact on the CPU based models. In the yearly license usage report [8] published by Macrovision [10], Softsummit [19], and the Centralized Enterprise Licensing User Group (CELUG) [3], etc, a general view of typical license and business models is given and analyzed and compared in the viewpoint of the ISVs and enterprises. It is evident, that the interests of software vendors and the enterprises are conflicting. For example, the per seat model is one of the most preferred pricing models for ISVs; while the enterprises will prefer the concurrent user model.

In the following, typical business/pricing models for licensing the software [8] are introduced.

- **Subscription:** Licenses are paid for with a recurring (often annual) fee to continue using the software. If the fee is not paid, the software stops working.
- **Perpetual:** Licenses are paid for on a one-time basis, giving the user the right to run the program as long as he/she chooses. It does not imply a right to upgrades, which are typically sold separately as part of a maintenance agreement or on a per-upgrade basis. Some vendors sell perpetual licenses on a term basis, which on the surface appears to be subscription based because the payments are spread out over time.
- **Concurrent User:** Software is licensed based on how many users may access the software simultaneously. Such license models are often used for business/enterprise software.
- **Seat (per machine/per server):** The software usage is restricted to a specific machine or server.
- **Per CPU or per CPU-core:** The software will be licensed to run on a specific CPU. With the wide adoption of the multi-core CPU technologies, the ISVs will make different license policies, that is, they will license the software per CPU or per CPU-core. Usually the enterprises will prefer the per CPU model.
- **Usage metric/pay-per-use;** The users will pay according to the real usage of their license.
- **Seat (named user):** In this model, each software license and corresponding usage rights are assigned to a specific person.
- **Financial metric based licensing:** License models that are based on varying business, usage or financial metrics, such as revenue, budgets, or cost of goods sold.
- **Custom contract:** For some business sectors, e.g., Electronic Design Automation (EDA) software, there is no price list, prices will be agreed upon individually in the customer contracts.

Similar to the licensing models the individual business/pricing models can be combined. There will be different business models with respect to different versions of the business software. e.g., Oracle is charging differently depending on different versions [29]. There are some new trends for the software license using and pricing according to the yearly license usage report [8], e.g., the subscription model will become more and more predominant method to license the software. The ISVs expect that the subscription model will increase their revenue in the future and their software will be more widely adopted.

3.3 Software as a Service

Software as a service (SaaS) is a rapidly growing business model of software usage and in consequence software licensing. In contrast to traditional models where users buy a perpetual-use license, SaaS users buy a subscription from the service publisher. Whereas traditional ISVs typically release new product features as part of new versions of software once in a few years, publishers using SaaS have an incentive to release new features as soon as they are

completed. There are several key characteristics of software delivered by SaaS which are identified in the report of IDC [38], including:

- Network-based access to and management of commercially available software
- Activities that are managed from central locations rather than at each customer's site, enabling customers to access applications remotely via the Web
- Application delivery that typically is closer to a one-to-many model (single instance, multi-tenant architecture) than to a one-to-one model, including architecture, pricing, partnering, and management characteristics
- Centralized feature updating, which obviates the need for downloadable patches and upgrades.

The property of the software as a service [32] licensing model leads to greater investment in product development under most conditions. This increased investment leads to higher software quality in equilibrium under SaaS compared to perpetual licensing. However, there are some weaknesses in this model, e.g., the prerequisite of accessing the software is the internet connection, and also sensitive information of the users will probably be stored at the SaaS provider side, thus a trust relationship has to be established. According to the predictions of Gartner [5], by 2012, at least one-third of business application software spending will be as service subscription instead of as product license. All leading business applications vendors (Oracle, SAP, Microsoft) and many web technology leaders (Google, Amazon) will promote this model, and the SaaS model of deployment and distribution of software services will become the mainstream use during the next five years. There are some companies which are adopting the SaaS models, e.g., NetSuite [11] offers subscription-based access to its enterprise resource planning (ERP), CRM, business intelligence software which is targeted toward small and medium-sized businesses. Salesforce.com [15] as one of the pioneers in deploying the SaaS, it provides the on demand customer relationship management solutions built on its infrastructure and the services will be delivered directly to users over the internet. The SaaS model is mostly attractive for smaller businesses because they are less willing to invest in large, expensive systems that they have to maintain.

3.4 License Scheduler

If the number of available licenses available for an enterprise is limited, e.g., due to the cost factor, it is necessary that these licenses are efficiently managed and highly utilised since even if an enterprise can apply for additional licenses from the ISV, it has to pay for the extra licenses. A local license scheduler could help scheduling the licenses of a site efficiently. However, while in most cases the local license management system provides information on the licenses already in use and still available there are no built-in queuing or reservation mechanisms. While an external scheduler might create an efficient schedule for the available licenses based on the users' requests, monitoring the use of the licenses and enforcing the schedule is difficult due to the usually encrypted communication between license server and application. Co-operations between license technology providers and license scheduler implementations could be a way to overcome this limitation. For instance, platform computing offers a product called LSF license scheduler [13] which is a local license scheduler restricted in a single administration domain and manages the license tokens instead of controlling the licenses directly. The current available number of licenses can be obtained by the FLEXnet manager. There are several license scheduling policies provided, e.g., fairshare, round robin, preemption. The licenses can also be checked out for the non-LSF jobs. In this way, the licenses can be scheduled and co-allocated with other resources/services. Dong et al. [34] developed a software sharing system in the grid environment which is not restricted to a single domain. The system adopts the constellation model for resource management and combines the sharing and scheduling of both hardware and software license resources. However, there is no support for SLAs and QoS in this system.

4 SLA Based Software Licenses and License Management

As introduced before, license management in Grid context is challenging. In the following sub-sections we will analyse the requirements and challenges for license management and present two solutions developed in European projects.

4.1 Requirements and challenges for license usage and license management in Grids

The different business/pricing models introduced before can be leveraged in Grid environments, however, the requirements and usage mode in the Grids should be considered. Obviously, the license models in Grids should be evolutionized in order to allow a smooth transition from the current practice dealing with the challenges from both technical and business/pricing aspects. The following list identifies these aspects in detail:

- **Different administrative domains:** License management may involve more than a single administrative domain. Therefore, issues like e.g., firewalls, remote usage control should be considered. Also, different usage policies might be defined for different administrative domains.
- **Transparent management:** Licenses should be transparently managed as part of the Grid job management.
- **Co-allocation of different resources:** Co-allocation of computing resources and software licenses should be supported. Software licenses should be co-scheduled together with other kinds of services or resources.
- **Remote license enforcement** Jobs may be executed remotely while the validity of the licenses has to be guaranteed at the same time.
- **Virtual organizations (VOs):** In the Grid VOs are often dynamically created and their members need temporal access rights for specific software suites from different domains, so flexible means of obtaining the temporal licenses should be provided.
- **Dynamicity:** The ability to suspend, preempt and resume the license use should be supported.
- **Interoperability:** License management should be integrated with the common Grid middlewares, e.g., GT4 [35], glide [4], UNICORE [21]. Licenses and license management should be built on standards instead of proprietary solutions.
- **Costs:** The scalability of the Grid influences the costs using the existing license models. Models like paying per-CPU, per-Seat, per-Job may quickly become expensive in Grid environments, concurrent floating licenses across the Grid are also too expensive for the software users [31]. Moreover, even if the price can be agreed upon, some issues remain, e.g., how the license usage will be audited and instrumented.
- **Support for workflows:** With the adoption of web services, service oriented architectures and BPEL, complex applications often are composed as workflows, which makes the license management more difficult. For instance, when executing a workflow, different applications may be used in different phases of the workflow. How to retrieve and reserve the right licenses for the applications in advance is one of the issues that need to be addressed.
- **Support for virtualization:** Licenses and license management for environments with virtualised resources or based on multicore technology is required.
- **Software is a service:** SaaS and on demand use as well as the utility pricing mechanisms should be considered both for the license models and license management in Grid environments.

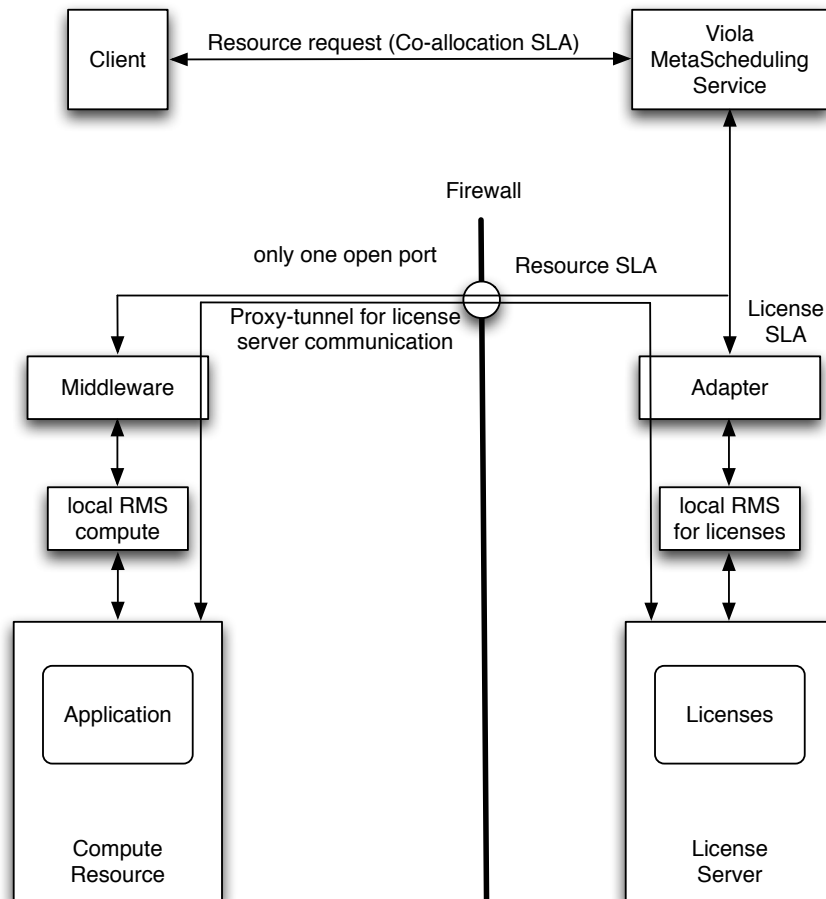


Figure 1: Basic approach of the BEinGRID project.

Considering the requirements for Grid environments, according to the 451 report [31], most of the enterprises want to have more flexible license models from the ISVs. It will be a great advantage for the enterprise if the licenses can be dynamically moved and managed in the global Grid. E.g., the EDA software licenses are bounded to some specific CPUs, machines today, while the enterprises hope that the ISVs of EDA software will support the Grid-wide licenses models so that the companies can run the software anywhere. However, the EDA vendors are reluctant to do so until today. Some major pharmaceutical ISVs also do not want to change their ways of licensing softwares. On the other hand, pharmaceutical companies may immediately benefit from using Grid resources to increase their computational power since many of the applications of the pharmaceutical sector are embarrassingly parallel and well suited for distributed environments with only best-effort network connectivity.

4.2 SLA-based Licenses and License Management

Recent R&D in two European projects [2, 18] shows that most of the requirements and challenges for licenses and license management in Grid computing can be tackled using a SLA based approach. In the preparatory process of defining the policies for software usage, which are governed by individual SLAs later, the policies and conflicts between the software vendors and the software users can be observed and reconciled. Negotiation is the preferred approach to create the SLAs between the license server and the user or the software entity acting on behalf of the user. These negotiation processes should be automatically executed based on a job description using JSDL and license description language (LDL) considering the large scale of the transactions and the scalability of the Grid infrastructures [33, 36]. WS-Agreement [30] and WS-Negotiation [22] are considered as the best suited existing or evolving negotiation protocols to create the SLAs between the respective resource management systems, license management systems and

the end users [37]. The definition of the term language used to represent and manage the licenses is currently under definition in the SmartLM project. The Job Submission Description Language [7] is already used in agreements as a term language for computational resources. The LDL describing license requirements is defined as an extension of JSDL. Such LDL includes terms like *life time of the license agreement*, *hardware environment the license might be used in*, *software features available through the license*, *license model and pricing model*, *compensation policies if the agreement is not fulfilled*.

4.3 Current Approaches to SLA-based License Management

On a European level currently two projects funded by the European Commission are addressing license issues in the Grid

- the BEinGRID project (Business Experiments in GRID) deals with licenses issues through a dedicated horizontal activity in one of its service clusters
- the SmartLM project that is in total focussing on a general solution for Grid-friendly software licensing for location independent application execution.

4.3.1 BEinGRID

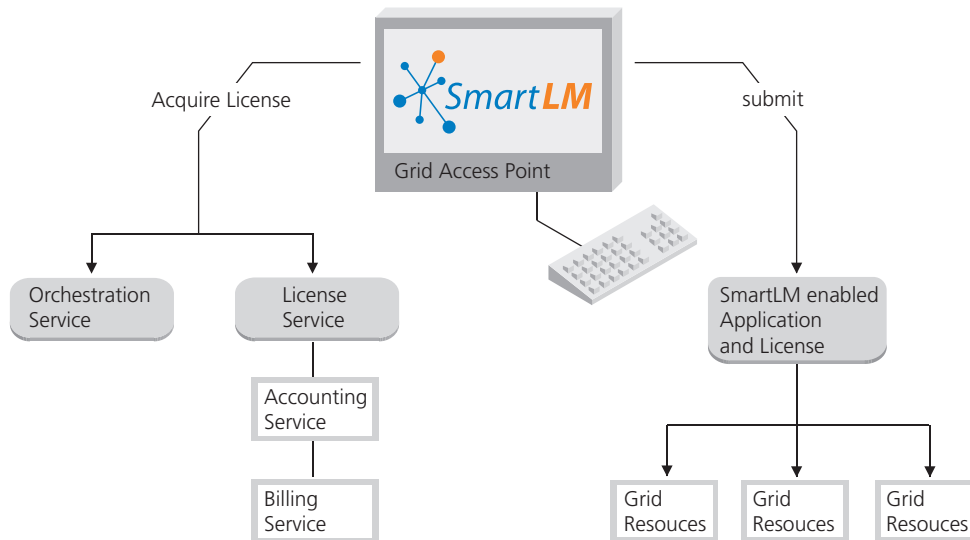
Figure 1 shows the initial basic idea of the solution implemented in BEinGRID [1]. A meta-scheduling service (MSS) [39] is responsible for the negotiation of reservations of computational services and license services. A local license resource management system is in charge of supporting flexible license models and policies. This component also supports the reservation of licenses for later use and allows queries about reservations made and current license usage. The adapter will connect to the local license resource management system implementing the API of the system on one end while supporting the WS-Agreement based negotiation with the MSS. Thus the MSS is able to negotiate agreements for the reservation of the required compute services and the appropriate license required for the execution of the protected application.

Other than the SmartLM solution, BEinGRID is targeting on enabling use of existing license technologies for Grids without addressing new license mechanisms or new business models. The approach is based on a proxy solution that transparently tunnels the communication between the remotely executed application and the FLEXnet server while at the same time making the FLEXnet server believe that it is talking to an application running locally. The current BEinGRID implementations differs from the initial approach since the MSS is not used for the negotiation of the co-allocation but the co-allocation is done manually by the user. However, the BEinGRID implementation provides additional services like improved security features, accounting and interfaces for billing. In addition to UNICORE and Globus Toolkit 4 the BEinGRID solution also supports the GRIA middleware environment.

4.3.2 SmartLM

In contrast to BEinGRID, SmartLM [18] is focussing on developing both licensing technology suitable for the use in distributed environments like Grid *and* - together with ISVs - new business models for the use of licensed software in Grid or SOA environments. These new business models are also relevant for environments where application service providers start delivering their compute resources to their customers through virtualisation. Similar, the growing number of clouds where service providers offer a completely virtualised computation environment may benefit from the developments since the classic license mechanisms and the related business models will not work anymore. The project starts with adapting codes of the three participating ISVs; ANSYS, INTES and LMS. In the second phase the project aims to attract other ISVs through its extensive dissemination and marketing activities. Figure 2 depicts the general idea of the SmartLM solution. A grid scheduler (MSS) is retrieving the Grid resources required by the user as described in his job. Once the appropriate resources have been identified, the local license service managing all licenses of a site is contacted to negotiate a SLA for the remote use of a license protected application. If both the requested compute resources and the license(s) are available they are reserved for the use by the requesting user. Like in the BEinGRID approach, a local license resource management system is in charge of the reservation of the licenses.

As said before, SmartLM is devoted to implement existing or evolving standards thus WS-Agreement is used for the creation of the SLAs and the project is contributing to the current work in the GRAAP-WG of the OGF on WS-Agreement-Negotiation. The resulting SLA is defining the terms of the application usage based on the local policies



SmartLM's distributed architecture

Figure 2: Basic approach of the SmartLM project.

and the privileges of the user. This SLA is then bundled by the orchestration service with the SLA for accessing the remote resources and transferred to the remote site where the job is executed. Since no communication is required between application and license server at run-time and the license SLA is created at the site of the user firewalls do not cause a problem. Moreover, the application may be migrated to another machine if the license SLA is migrated as well.

The application may run without any internet connection to the license server during run-time. However, if there is - at least temporarily - a connection to the site hosting the license server additional functionality of the new license mechanisms may be used, e.g. re-negotiation of the period the license is valid in case the application needs more time than foreseen by the user. Or, freeing a license that is no longer needed.

5 Conclusion

Software licensing and pricing is an important problem to be solved for a better acceptance of Grid infrastructures in productive environments with commercial applications. We present the state of art of software licensing and typical licensing and business and pricing models for managing software are analyzed. Further we identify requirements and challenges for using license protected software in the Grid. Ongoing work in two European projects aiming to overcome the limitations of existing license mechanisms in the Grid has been presented. The approach implemented in the BEinGRID project will become available in spring 2008 through the projects Gridipedia site. A prototype of the SmartLM solution including both new business models and the enabling technology for distributed environments will be available begin of 2009.

6 Acknowledgment

This paper includes work carried out jointly within the CoreGRID Network of Excellence funded by the European Commission's IST programme under grant #004265.

References

- [1] BEinGRID Open Source Grid Software Repository. <http://www.gridipedia.eu/index.php?id=702>.
- [2] BEinGRID Project. <http://www.beingrid.eu>, 2008.02.
- [3] Centralized Enterprise Licensing User Group. <http://www.celug.com/>, 2008.01.
- [4] Egee glite project. <http://glite.web.cern.ch/glite/>, 2008.02.
- [5] Gartner's top 10 IT predictions. <http://www.pcwelt.de/it-profi/englishnews/Hardware/145606/>.
- [6] IBM License Management. <http://www-306.ibm.com/software/awdtools/lum/sysrequirements.html>, 2007.06.
- [7] Job Submission Description Language WG (JSDL-WG). http://www.ogf.org/gf/group_info/view.php?group=jsdl-wg, 2007.3.
- [8] Key Trends In Software Pricing and Licensing (2006-07). http://www.softsummit.com/softsummit_knowledge_library_industry
- [9] License Asset Manager. <http://www.teameda.com/licenseassetmanager.html>, 2008.02.
- [10] Macrovision WebSite. <http://www.macrovision.com/>.
- [11] NetSuite Software. <http://www.netsuite.com/portal/home.shtml>.
- [12] Open iT License Management Software. <http://www.openit.com/>, 2008.01.
- [13] Platform LSF License Scheduler. <http://www.platform.com/Products/Platform.LSF.Family/Platform.LSF.License.Scheduler/>, 2007.06.
- [14] Reprise License Manager (RLM). <http://www.reprisesoftware.com/rlm.htm>, 2008.01.
- [15] Salesforce SaaS CRM Software. <http://www.salesforce.com/de/>, 2008.02.
- [16] Sentinel RMS. http://www.safenet-inc.com/products/sentinel/software_protection.asp, 2008.02.
- [17] Service Oriented Architecture. http://en.wikipedia.org/wiki/Service-oriented_architecture, 2008.02.
- [18] SmartLM project web pages. <http://www.smartlm.eu>, 2008.02.
- [19] Softsummit Conference. <http://www.softsummit.com/index.shtml>, 02.2008.
- [20] The license tracker. <http://www.licensetracker.ca/index.htm>, 2008.01.
- [21] Unicore Open Source. <http://www.unicore.org>, 2008.02.
- [22] Web Services Agreement Negotiation Specification. http://www.ogf.org/gf/group_info/view.php?-group=graap-wg, 2006.11.
- [23] *Flexnet Licensing 11.4 Programming and Preference Guide for Trusted Storage-Based Licensing*. Macrovision, 2006.
- [24] *Flexnet Licensing End User Guide*. Macrovision, 2006. Product Version 11.4, Document Revision 01.
- [25] *Flexnet Licensing for Java Programming Guide*. Macrovision, 2006. Product Version 11.4, Document Revision 01.
- [26] *Flexnet Licensing Programming and Reference Guide for License File-Based Licensing*. Macrovision, 2006.
- [27] *Getting Started With the Licensing Toolkit for License File-based Licensing*. Macrovision, 2006. Product Version 11.4, Document Revision 01.
- [28] *iFOR/LS Quick Start Guide*. HP, 2006. HP Part No, B2355-90108, June 1996.

- [29] Oracle software investment guide. Technical report, Oracle Corporation, 2007. <http://www.oracle.com/corporate/pricing/sig.pdf>.
- [30] Web Services Agreement Specification, 03 2007. http://www.ogf.org/gf/group_info/view.php?group=graap-wg.
- [31] Grid computing - the impact of software licensing. Technical report, The 451 Group, March, 2005.
- [32] Vidyanand Choudhary. Software as a service: Implications for investment in software development. *hicss*, 0:209a, 2007.
- [33] K. Czajkowski, I. Foster, and C. Kesselman. Agreement-based resource management. *Proceedings of the IEEE*, 93(3):631–643, 03 2005.
- [34] Xiaoshe Dong, Yinfeng Wang, Fang Zheng, Zhongsheng Qin, Hua Guo, and Guofu Feng. Key techniques of software sharing for on demand service-oriented computing. In Yeh-Ching Chung and José E. Moreira, editors, *GPC*, volume 3947 of *Lecture Notes in Computer Science*, pages 557–566. Springer, 2006.
- [35] I. Foster and C. Kesselman. Globus: A Metacomputing Infrastructure Toolkit. *The International Journal of Supercomputer Applications and High Performance Computing*, 11(2):115–128, 1997.
- [36] J. Li. *Strategic Negotiation Models for Grid Scheduling*. PhD thesis, Universität Dortmund, Informationstechnik, 2007.
- [37] Jan Seidel, Oliver Wäldrich, Philipp Wieder, Ramin Yahyapour, and Wolfgang Ziegler. Using sla for resource management and scheduling - a survey. In Domenico Thalia, Ramin Yahyapour, and Wolfgang Ziegler, editors, *Grid Middleware and Services - Challenges and Solutions*, volume 8 of *CoreGRID Series*. Springer, 2008.
- [38] Erin Traudt and Amy Konary. Software as a service taxonomy and research guide. Technical report, IDC, 06 2005.
- [39] Oliver Wäldrich, Philipp Wieder, and Wolfgang Ziegler. A meta-scheduling service for co-allocating arbitrary types of resources. In Roman Wyrzykowski, Jack Dongarra, Norbert Meyer, and Jerzy Wasniewski, editors, *PPAM*, volume 3911 of *Lecture Notes in Computer Science*, pages 782–791. Springer, 2005.