**Abstract**

Grid computing describes a computing model that distributes processing across an administratively and locally dispersed infrastructure to create virtual supercomputers at low cost. However, currently Grids are mainly employed within enterprises to connect internal divisions and business units. This paper attempts to explain why Grid market initiatives have failed. The explanation mainly focuses on the object traded in Grid markets. What is needed to extend Grid technologies beyond company borders is a set of mechanisms that enable users to discover, negotiate and pay for the use of Grid services on demand. This paper derives a roadmap for the design of market mechanisms based on a solid understanding of the technical possibilities. This roadmap underlines the need for a catalogue of market mechanisms to increase the impact of Grid markets in commercial settings.

**Keywords:** grid computing, service markets, market engineering, electronic markets

**Authors**

Dirk Neumann
(neumann@iism.uni-karlsruhe.de) is an Assistant Professor with the Institute for Information Management and Systems of the University of Karlsruhe, Germany. His research topics include market engineering, auction theory, automated negotiation, grid economics, and Peer-to-Peer economics. Currently, he is coordinating the EU-STRP project SORMA `Self-Organizing ICT Resource Management`

Jochen Stößer
(stoesser@iism.uni-karlsruhe.de) is a PhD student and research assistant at the Institute of Information Systems and Management at University of Karlsruhe (TH), Germany. His main research interest is in the design and prototypical implementation of market mechanisms for an efficient allocation of computer resources among self-interested agents.

Christof Weinhardt
(weinhardt@iism.uni-karlsruhe.de) holds the Chair for Information Management and Systems at the Department of Economics and Business Engineering at the University of Karlsruhe. His research area comprises e-Business, market engineering, P2P-based architectures, auction theory, matchmaking and agent-based bidding.

**Bridging the Adoption Gap – Developing a Roadmap for Trading in Grids**

**INTRODUCTION**

Grid computing is increasingly gaining traction in a number of application areas. It describes a computing model that distributes processing across an administratively and locally dispersed infrastructure. By connecting many heterogeneous computing resources, virtual computer architectures are created, increasing the utilization of otherwise idle resources (Foster et al. 2001). By the use of Grid technology, it is possible to set up virtual supercomputers via the connection of normal servers that cost no more than US$3,000 each. Adding and removing servers is simple, granting extreme flexibility in building up infrastructures. The best example of what Grid technology can achieve is illustrated by its prominent predecessor SETI@home: Over one million computers spread across 226 countries were connected to reach a processing power of 418.6 TFLOPS (Cirne et al. 2006). By comparison, the world’s fastest supercomputer, IBM’s BlueGene/L, has an estimated total processing power of between 280 and 267 TFLOPS, whereas Google’s search engine can muster between 126 and 316 TFLOPS estimated total processing power. The business case for Grids is further underlined by potential cost savings. It has been projected that Grids may decrease total IT costs by 30% (Minoli 2004). Thus, it is not surprising that Insight Research projects an increase in worldwide Grid spending from US$714.9 million in 2005 to approximately US$19.2 billion in 2010 (Anonymous 2006).

Currently, Grids are mainly employed within enterprises to connect internal divisions and business units. What is needed to extend Grid technologies beyond company borders is a set of mechanisms that can enable users to discover, negotiate and pay for the use of Grid services on demand. According to The451Group, a leading Grid research institute, the application of resource trading and allocation models is a crucial success factor for establishing commercial Grids (Fellows et al. 2007). Recent Grid offerings by Sun Microsystems and Amazon.com represent a first step in this direction. Sun introduced network.com, which offers computing services for a fixed price of $1 per CPU hour, while Amazon is currently launching comparable initiatives with its Amazon Elastic Compute Cloud (Amazon EC2) and the Amazon Simple Storage Service (Amazon S3).

Despite these first approaches, electronic marketplaces for Grid
resources have not yet taken off. Very few customers are using Sun’s network.com. Due to different legal frameworks, network.com was only offered to customers within the US. The adoption patterns in the US resemble those of the rest of the world – few customers have adopted Grid markets.3 What is the reason for this limited success of Grid markets? Almost every large computer hardware manufacturer like HP, Sun, or Intel has already worked on or at least considered the options for Grid markets, but still no Grid market has successfully been launched, creating a Grid adoption gap.

This paper attempts to explain why Grid market initiatives have failed. The explanation mainly focuses on the object traded on Grid markets. The problem with current markets for Grids is that they are designed purely as markets for physical resources. For example, Amazon’s Elastic Compute Cloud only aims to sell CPU hours. This type of market is by design not relevant for enterprise customers who have deadlines for executing jobs and who have no idea of how many resources are required to meet this deadline. As a result of the analysis in this paper, it is concluded that a single Grid market for physical resources such as CPU and memory is insufficient to ensure successful take-up of Grid computing across organizational boundaries. Instead, a set or catalogue of different marketplaces is needed to satisfy the diverse needs of different market segments. This paper begins the process of cataloguing the needed market mechanisms. Thus, this paper provides guidance for potential Grid market operators (e.g., Telecom companies or hardware vendors such as Sun Microsystems) in the choice of the market mechanisms needed to increase the impact of Grid markets in commercial settings.

The remainder of this paper is structured as follows. The second section discusses ways to define the trading object in Grid markets and thus forms the background for the discussion of market mechanisms. The third section explores whether one marketplace alone is sufficient for meeting the needs of Grid providers and users. A two-tiered market structure is proposed as a viable solution for structuring commercial Grids. The next section discusses the use of different market mechanisms for this two-tiered market structure and shows which mechanisms are most adequate for which kind of application. The final section concludes the paper with a summary and points to future work.

BACKGROUND DISCUSSION

Applications in a Grid environment can be deployed in two ways, either by directly accessing resources that are distributed over the network or by invoking a Grid service4 which encapsulates the respective resources behind standardized interfaces. These alternative ways of deployment give rise to different requirements for potential markets.

From a technical point of view, resources are simple to describe as there exists only a finite set of well-defined resources. A resource may be characterized by its operating system (e.g. Windows, Linux), number and type of CPUs (e.g. 4 * x86), memory (e.g., 128MB RAM), or any finite number of other attributes. The standardization of resources offers a simple way to describe them semantically. Glue Schema,5 for instance, provides a standardized vocabulary for characterizing computing elements and their status. This in turn facilitates resource discovery, as matchmaking within the finite domain space is straightforward.

Services on the other hand can be extremely difficult to describe. The service space is essentially infinite due to the myriad of variations in service design. For the description of complex application services (e.g., a virtual web server service ‘Apache on Linux’), domain-dependent ontologies may become necessary. In the case of ‘resource-near’ services, henceforth called ‘raw application services’ (for instance, computational services which use only CPU and memory), standardized languages such as the Job Submission Description Language (JSDL)6 exist. Nonetheless, the indefinite search space tremendously exacerbates service description and likewise service discovery.

Both resources and services are provided on the basis of Quality-of-Service (QoS) assertions, i.e. essentially guarantees of access to specific resources or services at specific times and under specific conditions. For resources, the QoS description is simple, as only standardized properties and the duration of resource access matter. For services, however, QoS is more difficult, as not only time aspects but also precision and accuracy of the services play a role. The definitions of precision, accuracy and further parameters depend on the individual service and cannot be standardized. This also has ramifications for monitoring. While monitoring resource access is relatively simple, the monitoring of complex application services becomes particularly demanding when services are intertwined.

Figure 1 shows the different aggregation levels of services and resources with respect to the layers of the Grid Protocol Architecture (Joseph et al. 2004). On top of this stack, the actual Grid application is located, such as a demand forecasting tool for Supply Chain Management or computer-aided engineering tools for simulations, etc. While these applications can be run directly on physical resources, they may access several complex application services, e.g., services for integrating, aggregating and statistically analysing vast amounts of data, to simplify development and deployment by shielding parts of the Grid’s complexity from the application. Complex application services are so diverse that they cannot reasonably be standardized. These
complex application services might in turn access raw application services which provide standardized interfaces for accessing various data sources and or computational services. These raw application services are resource-near services (such as storage, memory or computation services). Raw application services also comprise applications and software libraries, which can be standardized. These physical resources may be CPUs, memory, sensors, other hardware and software or even aggregated resources such as clusters (e.g., a Condor cluster) and designated computing nodes.

When relying on direct access to physical resources, executables and external libraries need to be transferred across the Grid. Typically, state-of-the-art Grid middleware only supports limited resource management functionality. In most cases, the middleware does not enforce policies concerning how many resources a job can consume. Only the local administrator can specify the degree to which resources can be shared. Trading physical resources is thus difficult to achieve by means of Grid middleware. Trading physical resources, however, is possible on the operating system level, which supports effective resource management. So called Grid Operating Systems, henceforth Grid OS (e.g., MOSIX7), support resource management on the OS Kernel level and are potentially available for setting up markets (Stößer et al. 2007). With Grid OS, applications need not be altered to be run on the Grid, as is the case when Grid middleware is being used.

From an economic perspective, Grid resource markets are promising for automation via an organized electronic market. There are standardized items for sale that potentially attract many buyers and sellers. However, complex application services have the disadvantage that demand is highly specialized and distributed across niche markets, such that only a few potential buyers are interested in the same or related application services.

**ONE MARKET FITS ALL?**

In this section, we explore whether it is sufficient to build up and operate a single market for Grids. Based on the previous background description, the answer to this question is straightforward: Designing one Grid market for all kinds of resources, from physical resources, such as processing power, memory and storage running on native platforms, to sophisticated virtual resources or application services that bundle and enrich such physical resources, seems inappropriate due to both technical and economical factors:

a. Technical factors: From the technical point of view, differences in the monitoring and deployment of services and resources make it very difficult to devise a generic system capable of supporting all kinds of resource and service trading. Furthermore, different deployment mechanisms impose different requirements on the market mechanism, as we will outline below.

b. Economic factors: From the economic point of view, market mechanisms need to achieve the following standard objectives in mechanism design (Stößer et al. 2007):

- **Allocative efficiency**: This is the overall goal of market mechanisms for Grid resource allocation. A mechanism is allocatively efficient if it maximizes the utility across all participating users (welfare or overall “happiness”).
- **Budget-balance**: A mechanism is budget-balanced if it does not need to be subsidized by outside payments.
- **Computational tractability**: The market mechanism needs to be computed in polynomial run time in the number of resource requests and offers.
- **Truthfulness**: This means that it is a (weakly) dominant strategy for users to reveal their true valuations to the mechanism.
- **Individual rationality**: A mechanism is individually rational if users cannot suffer a loss in utility from participating in the mechanism, i.e. if it is individually rational to participate.

As mentioned in the background discussion, trading physical resources and trading application services impose completely different requirements on the market: while physical resources are more or less commodities for which auction mechanisms seem to work well, (complex) application services are inherently non-standardized, thus making auction-like mechanisms inapplicable.

From this brief discussion, it can be seen that a one-size-fits-all market for Grids is not feasible from both the technical as well as from the economic perspective. Due to the heterogeneous properties of Grids, they can be divided into two different types of markets: resource-near markets for physical resources and raw application...
services on the one hand, and markets for complex application services on the other hand, spanning out what may be called a ‘two-tiered Grid market structure’ as depicted in Figure 2. These two classes of markets are analysed below in terms of their requirements for market structure.

Tier 1 markets

In the resource-near market for physical resources and raw application services, low-level resources such as processing power, memory and storage are traded. Demand in this market is generated by complex application services that need to be executed on these physical resources and raw application services. This setting poses special requirements for market mechanisms (Schnizler et al. forthcoming). However, as mentioned earlier, the requirements also depend on the way resources are deployed.

Deployment as physical resource. The technical requirements in markets for physical resources are the following:

- **Multi-attributes:** physical resources have quality attributes such as CPU speed, operating platform or bandwidth. Thus the mechanism needs to cope with multiple attributes simultaneously;
- **Bids on bundles:** generally, users require a combination of physical resources to execute an application (e.g., CPU and memory). If the mechanism does not account for bids on bundles, the user faces the risk of obtaining only one part of a bundle (the so called ‘exposure risk’). The market mechanism thus needs to support requests for bundles of resources;
- **Online mechanism:** the allocation of the mechanism needs to be made instantaneously, as the market assumes the role of an operating system scheduler. The mechanism thus needs to be lightweight such that it requires little computation time. Online mechanisms are crucial in the case where information such as release time or request processing time is only gradually released to the scheduler. The scheduler must be able to mitigate past decisions that prove unfortunate when new information enters the system. For example, facing a decrease in the performance of an application, the mechanism may be required to allocate additional physical resources in a timely manner;
- **Split-proofness:** in some scenarios, one might want the mechanism to treat small and large requests in a fair manner. The mechanism may need to be split-proof in the sense that users cannot improve their requests’ priority by splitting them into multiple parts and submitting them under different aliases; and
- **Merge-proofness:** the mechanism must be stable in the face of strategic users, who build coalitions to improve their priority. Thus the mechanism needs to assure that users do not have an advantage by merging their requests;

![Figure 2. Two-tiered market structure](image-url)
**Deployment as raw application service.** The technical requirements in markets for raw application services are the following:

- Multi-attributes: a virtual machine, for instance, may be characterized by the number of CPUs, share of memory, cache and bandwidth of the underlying physical resource. A simple computational service may need to provide a certain speed and accuracy;
- Bids on bundles: a user program may require multiple interdependent raw application services in parallel;
- Time constraints: when raw application services are traded, the market mechanism needs to take time attributes into account. The requesters need to specify their demand, so that the market mechanism can efficiently schedule the requests according to the availability of resources and price. This situation differs from the trading of resources, where the market mechanism executes requests and the corresponding applications upon availability;
- Coallocation: capacity-demanding Grid applications usually require the simultaneous allocation of several homogenous service instances from different providers. For example, a large-scale simulation may require several computation services to be completed at the same time. This is often referred to as ‘coallocation’. A mechanism for raw application services has to enable and control coallocations;
- Coupling: for some applications, it may be necessary to couple multiple raw application services into a bundle in order to guarantee that the services are allocated from the same seller and will be executed on the same machine; and
- Resource isolation: security and performance considerations lead to the requirement of resource isolation, i.e. that a specific raw application service can only be instantiated once at any given time.

**Tier 2 markets**

Along the lines of the two-tiered market structure, complex application services can be decomposed into several raw application services that can in turn be translated into the physical resources required to execute these services. For instance, some complex application service might require a basic XML transformer service, which in turn needs processing power, memory, storage etc. Buyers in such a market request a complex application service; the provider of this complex application service, the service integrator, is responsible for obtaining the required raw application services and physical resources in turn, thus hiding parts of the Grid’s complexity from the buyer. Such a hierarchical masking of complexity seems to be an appropriate approach since service requesters typically have no insight into the resources the complex application service will consume (Eymann et al. 2006).

The requirements of complex application services for market mechanisms are totally different than for resource-near markets. Complex application services are rarely used by two different companies; hence creating competition via an auction mechanism does not make sense. Instead, the market for complex application services faces the difficulty of finding a counterpart that offers the exact capabilities needed to execute the application. As the following requirements suggest, the market mechanism is more ‘search-oriented’, in terms of the need for bilateral or multi-lateral negotiation protocols:

- Multi-attributes: (see above);
- Workflow support: To support complex services, distributed resources such as computational devices, data and applications need to be orchestrated while managing the application workflow within Grid environments. The market mechanism needs to account for this during design time and run time of the workflow;
- Scalability: scalability considers how the properties of a protocol change, as the size of a system (i.e. the participants in the Grid) increases; and
- Coallocation (see above).

As pointed out in the introduction, resource-near Grid markets are not a viable solution for enterprises which typically have to run time-critical applications. However, applications in academia are usually less time dependent. As such, resource-near markets would be a viable business model for such settings; the users have to wait until the queued jobs are executed but clearly, the issue of payment is controversial in academia. Even for the EGEE Grid, billing and payment will soon become an issue as demand exceeds supply. It seems that resource-near markets will soon become an adequate model for academic Grids such as EGEE or D-Grid, the German Grid initiative.

Grid markets that will be widely accessed by enterprises need to be of the form of markets for complex application services. Consider a manufacturer interested in executing a computer-aided engineering application and deploying it on a computationally intense platform. This complex application service showcases a very specific service which is likely to be demanded by a single requester only. To accommodate this complex application service, the service must be decomposed into its constituent raw application services and the required physical resources. Integrators – companies such as EDO2, GigaSpaces, etc. that specialize in aggregating and disaggregating services into physical resources – are needed to facilitate this decomposition process. The specialization stems from experience, allowing the identification of service needs by comparing each service request with similar service requests in the past, where
similarity is established in terms of algorithms, data structures and sizes, etc. Telecommunication companies and hardware producers seeking to virtualize IT infrastructures naturally have the interest and competency to become Grid services integrators. Table 1 summarizes this discussion.

A ROADMAP FOR GRIDS

In previous sections, we argued that Grids do not require something like a single global market where all Grid requests and supplies are aggregated, but a more complex two-tiered market structure. Figure 2 summarizes this (meta-)market structure. Applications demand the execution of several complex application services in Tier 2 markets. In resource-near markets (Tier 1), complex services request physical resources either plainly deployed or accessed via service interfaces. Integrators assume the responsibility for mediating between requesters unaware of their resource needs and the needed resources.

Based on the preceding discussion, we set up a taxonomy below of known market mechanisms that support different types of Grid applications. This taxonomy is conceived as a roadmap for further Grid market developments to help bridge the Grid adoption gap.

Market mechanisms for trading physical resources

For Grids where physical resources (but not services) are traded, no time restrictions apply. Usually, the resources themselves are not traded, but rather shares of computing units (e.g., nodes) are. The idea is that bidding determines the share a user receives from the pool of available nodes. The more resources a user obtains, the faster the application is completed. The following mechanisms have been proposed for this setting:

- Fair Share (Kay and Lauder 1988): In the Fair Share mechanism, all users get the same share of the respective resource. This share is dynamically adjusted as new users enter the system or existing users leave.
- Proportional Share (Lai et al. 2004): In contrast to the Fair Share mechanism, with Proportional Share the shares can differ across users to reflect each user’s priority. This priority may be determined dynamically based on the users’ bids for resources: If there are \(n\) users in the system and user \(i\) has submitted a valuation of \(v_i\), then \(i\) will receive a share amounting to \(v_i/\sum_{j=1}^{n} v_j\).
- Pay-as-bid (Bodenbenner et al. 2007, Sanghavi and Hajek 2005): The pay-as-bid mechanism operates in the same settings as Fair Share and Proportional Share but is specifically designed so as to induce users to truthful reporting of valuations. It has been shown that pay-as-bid improves on the prominent Proportional Share mechanism as regards efficiency and provider’s revenue (Bodenbenner et al. 2007).

All three mechanisms, however, share a common drawback: they can only be used in scenarios where one resource provider serves several consumers, that is, there is no competition among providers. In cases where all resources are under fully centralized control, this condition is unproblematic. However, the idea in Grids is to cross administrative boundaries. Consequently, there is a need for market mechanisms that support multiple resource providers.

Market mechanisms for trading application services

Dividing the service market into two parts – raw and complex application services – is too simplistic, as the timing of demand for services has not yet been considered. This timing is determined by the application itself and depends on the task the application is performing. We use the term ‘application model’ as a characterization of the processing mode of the application. This encompasses in particular the workload of the application as well as the interaction model between applications and the Grid middleware virtualizing the execution platform. Depending on the application

<table>
<thead>
<tr>
<th>Table 1. Types of grid markets</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Market level</strong></td>
</tr>
<tr>
<td>Deployment</td>
</tr>
<tr>
<td>Description languages</td>
</tr>
<tr>
<td>Time limits</td>
</tr>
<tr>
<td>Application areas</td>
</tr>
<tr>
<td>Requirements</td>
</tr>
</tbody>
</table>
model, different requirements upon the market mechanisms emerge:

- Batch applications (e.g., data mining) are characterized by a planned execution and expected termination time. Execution is serial and resource demands depend on the parameters, such as the size of input data.
- Interactive applications (e.g., online data analysis) are applications that require services on demand, depending on the interactions with users. In contrast with batch applications, it is not possible to plan the execution and expected termination time of interactive applications far in advance, so unpredictable peaks of requests can occur within a short time.
- Task-oriented applications are dynamically composed from (sub-) tasks to build more complex tasks. Service demand depends on the (work-) flow of requests from multiple users (e.g., a bank transaction system).

Most of the market-based approaches relate to batch applications. Batch applications are comparably easy for two main reasons. First, there is no need to consider a whole workflow with different resource demands on each echelon of the workflow. Second, the time to determine the allocation can be relatively long; immediacy is not essential. Thus, complex resource allocation computations can be performed without hampering the whole application due to latency times devoted to the calculation of the optimal allocation. Furthermore, most of the practical market-based Grid prototypes consider only one single resource type (e.g., CPU only) and thus make use of standard auctions (e.g., English auction). In applications other than pure number crunching, those auction types are inadequate as more than one object (e.g., memory) is required at the same time.

**Market mechanisms for raw application services.** As mentioned above, the market mechanisms for raw services depend on the application model. Mechanisms that we identify as being adequate for batch applications are:

- Multi-attribute Combinatorial Auction (Bapna et al. 2006): In the model of Bapna et al., multiple requesters and providers can trade both computing power and memory for a sequence of time slots. First, an exact mechanism is introduced. By imposing fairness constraints and a common valuation across all resource providers, the search space in the underlying combinatorial allocation problem is structured so as to establish one common, truthful price per time slot for all accepted requests. Additionally, this introduces a linear ordering across all jobs and time which reduces the complexity of the allocation problem, which however still remains NP-hard.13 To mitigate this computational complexity, a fast, greedy heuristic is proposed at the expense of both truthfulness and efficiency;
- Multi-attribute Exchange (Stößer et al. 2007): The model of Stößer et al. extends the model of Bapna et al. (2006) in that it accounts for strategic behaviour not only on the demand side of the market but also among resource providers. Stößer et al. design a truthful scheduling heuristic for Grid operating systems that achieves an approximately efficient allocation fast. A greedy heuristic is employed to solve the scheduling problem. It is complemented with an adequate pricing scheme which assures that reporting truthfully is a dominant strategy for resource requesters and payments to resource providers are approximately truthful;
- MACE-mechanism (Schnizler et al. forthcoming): Schnizler et al. elaborate a Multi-Attribute Combinatorial Exchange (MACE). Users are allowed to request and offer arbitrary bundles of computer resources and can specify quality attributes of these resources. The scheduling problem in this combinatorial setting is NP-hard, and the pricing scheme of MACE yields approximately truthful prices on both sides of the market. Due to the NP-hardness of the problem, the mechanism is not applicable to large scale settings and interactive applications that require the immediate allocation of resources;
- Combinatorial Scheduling Exchange (AuYoung et al. 2004): The Bellagio system also implements an exact combinatorial exchange for computing resources (AuYoung et al. 2004). Its pricing is based on the approximation of the truthful Vickrey-Clarke-Groves prices proposed by Parkes et al. (2001). As an exact mechanism it shares the computational drawbacks of MACE; and
- Augmented Proportional Share (Stoica et al. 1996): One major drawback of Proportional Share as proposed by Lai et al. (2004) is that users do not get any QoS guarantee. Generally, users will receive a larger or smaller share than required and constantly need to monitor their resource needs and the actual allocation. To mitigate this deficiency, Stoica et al. propose an extension of the Proportional Share mechanism so that users can request and receive a fixed share of the resource. Thus, this approach inherits the benefits of the simplicity of Proportional Share and resource reservation if required.

For interactive applications it is impossible to predict demand for raw application services. Thus, market mechanisms need to allocate these services continuously. This can be realized by frequent call mechanisms, where bids are collected for a very short time span and cleared immediately. This requires that the mechanism is solvable in few milliseconds. The greedy approaches of the Multi-attribute Combinatorial Auction Heuristic (Bapna et al. 2006) and the Multi-attribute Exchange
(Stößer et al. 2007) are capable of approximately solving the combinatorial allocation problem in the required timely manner. Alternatively, the mechanism could be a true online mechanism, which allows the real-time job submission to available resources (e.g., nodes or clusters). A third way is to introduce a derivative market to hedge against the risk of supernormal resource demand. Adequate market-based schedulers for interactive applications comprise:

- **Decentralized Local Greedy Mechanism** (Heydenreich et al. 2006, Stößer et al. 2007): This mechanism is designed to provide a scalable and stable auctioning protocol. It operates in an online setting where jobs arrive over time. Its aim is to schedule these jobs so as to minimize the overall weighted completion times. This scheduling mechanism is complemented by a pricing rule which induces the users to truthful behaviour;

- **Augmented Proportional Share** (Stoica et al. 1996): The shares which are allocated to the users and the resulting prices can be adjusted very efficiently as new requests are submitted or requests are finished. Moreover, users can choose their risk by either fixing the price – and receiving a corresponding share which may or may not correspond to the true demand of their interactive application – or by fixing their share and having to pay the corresponding price, which may or may not match their valuation; and

- **Derivative Markets** (Kenyon and Cheliotis 2002, Rasmusson 2002): Derivative markets perform two basic functions, hedging and speculation, which become essential in the context of non-storable goods such as Grid resources. This generates early price signals which support capacity planning for both buyers and sellers. Rasmusson (2002) proposes the use of options to price network resources. Kenyon and Cheliotis (2002) follow the same idea by proposing option contracts for network commodities such as bandwidth. They elaborate a pricing scheme for options which specifically takes into account aspects of the underlying network topology that may affect prices, e.g., the existence of alternative paths. Options cannot only be used for hedging purposes but also for speculation (arbitrage) which contributes liquidity to the market.

The requirements for market mechanisms that support **task-oriented applications** are very demanding, as all constituents of the workflow need to be allocated simultaneously, since otherwise the application cannot be fully executed and is of no value to the user. Currently, there are only bargaining protocols available that guide the users in their search for all components of the workflow, e.g., SNAP: The Service Negotiation and Acquisition Protocol (Czajkowski et al. 2002). SNAP supports the reliable management of Service Level Agreements (SLAs). In particular, SNAP supports the process of negotiating simultaneously across multiple resource providers, a key requirement for complex and resource-demanding services.

**Market mechanisms for complex application services.** Trading complex application services is very demanding as there are not many providers and requesters. Currently, there is not much research available that aims to develop market mechanisms for trading complex application services. Hence, the market mechanisms for batch, interactive and task-oriented applications do not differ substantially. In particular, the MACE-mechanism (Schnizler et al. forthcoming), the Bargaining Protocol (Czajkowski et al. 2002) and the Decentralized Local Greedy Mechanism (Heydenreich et al. 2006; Stößer et al. 2007) might be appropriate for trading such services. Furthermore, for all applications, the licensing model of software-as-a-service is applicable. It refers to take-it-or-leave-it pricing, where the vendor sets the price and the users decide whether or not to purchase.

Table 2 summarizes the existing market mechanisms for each application model.

<table>
<thead>
<tr>
<th>Application model</th>
<th>Physical resource</th>
<th>Raw application service</th>
<th>Complex application service</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Batch</strong></td>
<td>Fair share</td>
<td>MACE</td>
<td>MACE</td>
</tr>
<tr>
<td></td>
<td>Proportional share</td>
<td>Multi-attribute combinatorial auction</td>
<td>SNAP</td>
</tr>
<tr>
<td></td>
<td>Sanghavi-Hajek</td>
<td>Augmented proportional share</td>
<td>Software-as-a-service</td>
</tr>
<tr>
<td><strong>Interactive</strong></td>
<td>Decentralized local greedy mechanism</td>
<td>Multi-attribute auction</td>
<td>SNAP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Augmented proportional share</td>
<td>Software-as-a-service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Derivative markets</td>
<td>SNAP</td>
</tr>
<tr>
<td><strong>Task-oriented</strong></td>
<td>SNAP</td>
<td></td>
<td>Software-as-a-service</td>
</tr>
</tbody>
</table>
CONCLUSION

This paper argues that, despite its promising features, the technology of Grid computing has not yet been adopted by enterprises due to an absence of adequate business models. Grid markets may help to overcome this obvious ‘adoption gap’. The paper attempts to derive an economically sound set of market mechanisms based on a solid understanding of the technical possibilities.

In the background section, we analysed the characteristics of the object traded in Grids. This trading object is closely associated with the deployment of software applications. The deployment directly on physical resources or via raw application services has major ramifications for the trading object and consequently for the requirements for market mechanisms. Physical resources are essentially commodities, whereas application services can be both standardized commodities (raw application services) and unique entities (complex application services).

Based on this analysis, we derived a two-tiered market structure along the distinction between physical resources and application services, where each tier demands different market mechanisms. The first tier comprises the markets for physical resources (e.g., CPU, memory) and raw application services. The second tier comprises the markets for complex application services.

We then presented and classified existing Grid market mechanisms according to this market structure. At the core of this paper, we argue that there is no single market that satisfies all purposes. Reflecting the distinct requirements of different application models (physical resource vs. raw application service vs. complex application service) and deployment modes (batch vs. interactive vs. task-oriented), a catalogue of coexisting market mechanisms is needed. Thus, in order to overcome the adoption gap, Grid market operators such as Sun Microsystems and Amazon must deploy multiple market mechanisms via a Grid market platform to satisfy the needs of their enterprise customers, or else integrators must step in to bridge these niche markets.

In essence, this paper suggests several intriguing research avenues. Further research is needed to specify the properties of both the various application scenarios and the necessary market mechanisms. Since existing market mechanisms represent rather theoretical constructs, they need to be deployed in real-world settings to verify the underlying models and analytic results. However, existing mechanisms only represent a first step towards filling the identified market structure and satisfying the various requirements. This basic catalogue needs to be extended by means of improved and new mechanisms in order to further enhance efficiency and to account for the strategic dimension inherent in Grid markets. And, of course, the commercial adoption and adaptation of Grid markets requires an analysis of the supply side as well as of the demand side, which was our focus in this paper. An in-depth analysis of the supply side would address questions such as the following: ‘What are sustainable business models for companies that provide market platforms for trading Grid services or resources?’, and ‘How large is each of the individual markets within the two-tiered market structure?’ Chargeable Grid services and sustainable business models are not the final or only answers to the Grid adoption problem. Other factors, such as trust in the security and reliability of Grid technology, are also relevant. Yet better technology alone will not ensure widespread adoption of the Grid: sound market mechanisms are also necessary.

ACKNOWLEDGEMENT

This work has partially been funded by EU-IST programme under grant 034286 ‘SORMA-Self Organizing ICT Resource Management’.

Notes
3. Quite recently, Sun has been facilitating access to customers in 25 countries. It remains to be seen if the removal of legal concerns will result in broader adoption.
8. We use the term ‘market structure’ to denote the configuration of marketplaces.
9. It should be noted that there could be \( \pi \)-intermediate markets. We consider only the extreme cases, as they exhibit different characteristics.
13. Informally, for a complex/large problem instance, the allocation problem is not solvable with justifiable effort.
14. Software-as-a-service refers more to model of software delivery where a service provider (e.g. SAP) offers to requesters
applications that are specifically implemented for one-to-many hosting.

References