Association for Information Systems
AIS Electronic Library (AISeL)

ECIS 2006 Proceedings

European Conference on Information Systems (ECIS)

2006

Automated bidding for trading grid services

Dirk Neumann
University of Karlsruhe, dirk.neumann@iw.uni-karlsruhe.de

Steffen Lamparter
Universität Karlsruhe (TH), sla@aifb.uni-karlsruhe.de

B. Schnizler
University of Karlsruhe (TH), schnizler@iw.uni-karlsruhe.de

AUTOMATED BIDDING FOR TRADING GRID SERVICES

Neumann, Dirk, University of Karlsruhe (TH), Information Management and Systems, Englerstr. 14, 76131 Karlsruhe, Germany, dirk.neumann@iw.uni-karlsruhe.de

Lamparter, Steffen, University of Karlsruhe (TH), Institute of Applied Informatics and Formal Description Methods, Englerstr. 11, 76131 Karlsruhe, Germany, sla@aifb.uni-karlsruhe.de

Schnizler, Bjoern, University of Karlsruhe (TH), Information Management and Systems, Englerstr. 14, 76131 Karlsruhe, Germany, schnizler@iw.uni-karlsruhe.de

Abstract

For decades markets have been proposed for allocating computer resources in distributed systems like the Clusters or Grids. Nevertheless, none of these approaches has made it into practice. The reasons for the adoption failure of markets are manifold. One reason that has been hitherto rarely discussed is the bidding process. Theoretic approaches assume that the bidders know how to derive their bids exactly. This does not only include the specific computer resource, which is needed at some time in the future, but also the price. This assumption simplifies reality and can thus not contribute to the development of markets in Grid. What is needed to establish a prospering market for Grid resources are rules how to conduct the bidding. Since demand and supply are extremely dynamic in computing resources, manual bidding is too slow to accommodate abrupt shifts in demand or supply. This paper introduces a policy based autonomous agent approach for automated bidding. By means of the policies resource providers and consumers can specify the way how they trade Grid resources (e.g., resource isolation definitions, security specifications).

Keywords: Grid Market, Automated Bidding, Ontologies

1 2 INTRODUCTION

In the mid 1990s the term Grid was introduced for describing a distributed computing infrastructure for advanced science and engineering. Since then, considerable attention has been spent on constructing such Grid infrastructures (e.g., Globus, Unicore) that provide a *"dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities"* (Foster and Kesselman 2004). In the beginning, Grids resource allocators are usually organized as batch queuing resource management systems. Computational jobs are submitted to the resource management, which allocates and schedules them in the order they arrived. With the number of possibilities the infrastructure offers also the demand for computational resources (in particular for particle physics, automotive simulations, etc) exploded. With demand exceeding supply, the resource management systems embodying batch queuing systems have a problem to assign resources to the right jobs. Since it is no longer possible to accomplish all tasks, decisions must be made to include jobs on the one side and reject jobs on the other side.

In this context, the use of value-based scheduling mechanisms has been proposed. In essence, value based mechanisms require that to any job request or resource provision values or reservation prices are attached. With those values attached, a value maximizing allocation of resources can be accomplished. From other domains (e.g. securities trading), markets have proven to work well, as they create competition, which affects the users to reveal their true valuations. The use of markets in distributed computing, however, is not a new idea. Since the 60s researchers have attempted to make use of markets (Huberman 1988). Nonetheless, none of those attempts were adopted area-wide.

Recently, the idea of using markets for the Grid was revitalized by many researchers. The confidence why markets would now find their way into practice stems from many advances in the field of market design and Grid computing (Shneidman, Ng et al. 2005). While earlier attempts referred to quite simple market mechanisms (e.g. English auctions), which are unable to accommodate the needs of the Grid (e.g. multi-attributes, complementarities among the attributes), modern approaches refer to more complex schemes such as combinatorial auctions and exchanges. In addition to those market design issues, the Grid middleware is gradually improving allowing for market based resource allocations. In the last years prototypical solutions that implement market-based Grid systems have been developed (AuYoung, Chun et al. 2004).

Nevertheless, none of those newly developed systems has made it into practice. The reason why markets could solve

the resource allocation problems, but currently do not, are manifold (Shneidman, Ng et al. 2005). Among the reasons that are rarely discussed stands the bidding process. In essence, theory typically assumes that the buyers and sellers have a specific valuation for their resource demand and supply and, furthermore, are aware of this valuation. In practice, the bidding process is a deliberation process, which incurs (sometimes substantial) transaction costs for acquiring information, defining the strategy and lastly to enter the bid into the Grid system. Even though these transaction costs are low, the number of transactions may be high due to the dynamic nature of computing. More importantly, changes in demand can occur from one second to the other (e.g. for example in the presence of a sudden overload, capacity is needed right away to maintain an operating system); too quick for a human trader to enter a bid manually. Hence, there is a need not only for decision support, but also for automated bidding. Hitherto, approaches for automated bidding are very limited (Dumas, Aldred et al. 2005). The contribution of this paper is to provide an agent-based bidding process that supports the entire bid generation process for Grid resources by relying on rules, which are expressed as policies.

The remainder of the paper is structured as follows: Section 2 states the problem of bidding in more detail and derives a need for automated bidding. Section 3 depicts the general architecture of a Grid Market with integrated automated bidding components. Section 4 describes the specifications and the mechanisms how to derive bids automatically. Section 5 concludes with a summary and an outlook for future research.

3 PROBLEM DEFINITION

Today's resource management systems in Grids have recognized the need of expressing values by including user priority, weighted proportional sharing, and service level agreements that set upper and lower bounds on the resources available to each user or group (Irwin, Grit et al. 2004; LSF 2005). Maximizing the utility (i.e. the sum of valuations), however, is only possible if the resource manager knows the attached valuations or the exact relative weights, respectively at any point of time. Knowing the valuations at any time is a very demanding requirement, as users typically have no incentive to report decreases in their valuation, because they loose priority and correspondingly value by not getting their computation done.

Hence, value-oriented approaches are not sufficient per-se to achieve an efficient solution. Only if all participants are willing to report their priorities and values honestly, these algorithms (e.g. Proportional Share (Lai 2005)) will work well. This is where markets for Grid enter the discussion. Markets have the ability to set the right incentives for users to reveal their true valuation as well as for resource owners to provide those resources that are scarcest in the Grid. With the introduction of prices, incentives will be given to the users to substitute the scarce resource (e.g. number of CPUs) with less scare resource (memory). For instance, a fixed pricing scheme which requests 10 € for one CPU and 1 € for memory, sets the incentives to reduce the number of used CPUs in favor of the cheaper memory.

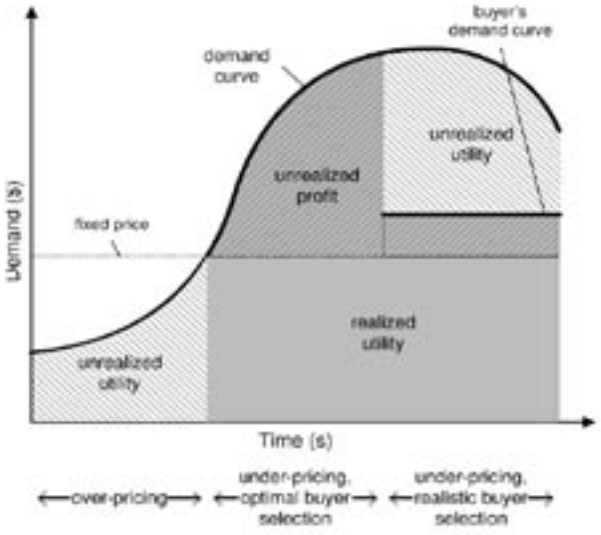

Figure 1: Fixed Pricing (Lai 2005)

A fixed pricing scheme, however, is not enough to achieve an efficient allocation: Suppose the fixed price of a resource provider as shown in Figure 1 (c.f. Lai 2005). Demand changes over time – this will be depicted by the parabola – without loss of generality the costs of supplying resources are assumed to be zero. If demand is below the fixed-price (left end of the graph), no resource will be requested. This is because the value for the resource, represented by the demand curve is

below the price. As a consequence, there is a loss of utility, which is denoted by the area below the demand curve. In the middle part of the Graph there could be more buyers willing to pay the fixed price, as their demand exceeds the price. If the seller allocates the resource efficiently to the user who values the resource most, there will be unrealized profit for the seller indicated by the striped gray area, which refers to the difference between the demand curve and the fixed price.

For resource providers the problem is to set the price for resources that accommodates demand as good as possible to extract a large portion of consumer's value (area below demand curve). By submitting bids, the resource providers can post their price expectations to the market. Those bids must be accurate with respect to the resources available and to the price – this requires the integration with the resource accounting systems. In analogy, the resource users need to express their valuations for certain resources at certain point of time via bids. The value for resources thereby is reflected by the value of the job that can be performed with the resources. The generation of bids thus also needs the integration with the resource accounting system and with the Grid application.

Due to the inherent dynamism in computer resource demand and supply, those bids need to be automatically generated. Manual bid generation would be too slow to be effective. For instance, overload situations occur suddenly, where manual reaction time in generating bids is not fast enough to maintain an operational system. Approaches to automate bidding, which incorporate more complex strategies for Grid, do currently not exist. This paper is unique as it provides a rule-based approach. In the next section, the necessary components for an automated Grid market are briefly introduced, before in the subsequent section the reasoning of the automated bidding is described.

4 MARKET-ORIENTED GRID ARCHITECTURE

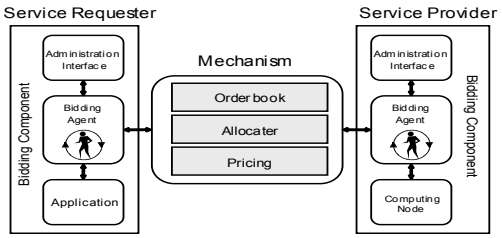

Figure 2: Grid Market with integrated automated bidding components

Figure 2 depicts the general architecture of a Grid market including automated bidding components. In essence, the innovation in this architecture pertains to the Grid market and to the middleware extensions, which are necessary to access the market. Resource sharing is technically realized by standard Grid middleware such as Globus Toolkit[1] and is thus not in the center of attention. Following the Globus framework, the sharing of resources is not realized by sharing physical resources (e.g. CPU), but by providing services as aggregated resources (e.g. CPU cycles).

Within this architecture, the market consists of service requesters (e.g. scientists running a simulation application), service providers (e.g. computer centers with idle resources), and a market mechanism capable of aggregating bids from resource providers and users and subsequently determining an outcome. Both, service requesters and provider, comprise a bidding component which issues bids to the market. In the following, the market components are more thoroughly described.

4.1 Bidding Component

As illustrated in Figure 2, a bidding component requires three elements: A *Bidding Agent* that generates bids and forwards them to the market mechanism, an *Administration Interface* that allows the control and configuration of the agent and either an *Application* in case of a service requester or a *Computing Node* in case of a resource provider.

Administration Interface: The Administration Interface can be used to specify user policies. These policies are either entered manually via a human control interface or they are imported from other sources in a company. For instance, considering general governance and compliance guidelines, which reflect a company's strategies and goals are crucial when generating a bid. Policies are mostly static and do not change frequently. In the context of Grid, policies define

general rules specifying when a bid has to be generated, how the bid should look like and under which conditions a certain service offer or request is suitable. For instance, a policy may state that the communication with a particular service must be encrypted or that services allocated at night are preferred to allocations during the day.

Application: In contrast to the static nature of policies, Applications provide runtime data of the requester's system. Based on this data, service requests are initiated including detailed service type requirements (e.g. a storage service) and quality of service constraints (e.g. a storage service with at least 300GB free space for four hours). It is assumed that approximate quality and time constraints of the requested services can be determined using prediction models (e.g. Kee, Casanova et al. 2004).

Computing nodes: On the provider side, Computer nodes host a set of Grid services (e.g. a computation service or a storage service) and provide information about their current quality characteristics and availability. The node includes a resource manager that provides runtime data about scheduling, allocating, and monitoring of local resources. Furthermore, this resource manager makes use of prediction models in order to determine the approximate future availability of the underlying computational resources.

Bidding Agent: A Bidding Agent implements a bid generation process which is based on policies as well as runtime data and is responsible for submitting bids to the market mechanism. Furthermore, the agent interacts with the market mechanism in order to gain status information (e.g. the highest price of a service) which influences its bidding decisions. The agent specifies all attributes of an order autonomously (e.g. the price or the bidding time).

The Bidding Agent integrates policy as well as runtime information via an ontology. Ontologies feature logic-based representation languages that come with executable calculi. This is required by the agent in order to query and reason about this information during run-time. To capture system information as well as policies we rely on the Web Ontology Language (OWL) and a decidable fragment of the Semantic Web Rule Language (SWRL) defined in (Motik, Sattler et al. 2005). The bids generated by the agent are encoded using the standardized agreement language WS-Agreement (Ludwig, Dan et al. 2004). In the following, we use the Description Logic abstract syntax as defined in (Baader, Calvanese et al. 2003) to formalize OWL expressions and standard rule syntax to define SWRL-rules.

4.2 Market Mechanism

Central for the market architecture stands the market mechanism component which implements a trading mechanism for Grid Services. The component is defined as a Web service in order to provide interoperable invocation facilities to the bidding agents. The market service interface is specified by means of the Web Service Description Language (WSDL) and comprises a list of provided operations including messages that are accepted and returned. For instance, the interface provides operations to retrieve status information (e.g. the current highest bid) and to submit bids.

The underlying institutional mechanism of the market component is a multi-attribute combinatorial exchange (MACE), as proposed by (Schnizler, Neumann et al. 2004). MACE allows multiple bidding agents (representing service requesters and providers) the simultaneous submission of bids on heterogonous services expressing substitutes (realized by XOR bids) and complements (realized by bundle bids). Furthermore, the mechanism is capable of handling cardinal attributes. For instance, a resource consumer can bid on a bundle consisting of a computation service and a storage service. The computing service should have two processors. Each processor should have at least 700MHz and the storage service should have at least 300 GB of free space.

Bids – encoded as WS-Agreement offers – are submitted to an order book which can be traced by bidding agents. Regularly (in case of a call market) or continuously (in case of continuous matching), the auctioneer determines an allocation (winner determination) and the corresponding prices.

The objective of the *allocation* process in MACE is the maximization of social welfare, i.e. the difference between the buyers' valuations and the sellers' reservation prices. The problem is formulated as a linear Mixed Integer Program (MIP) and, thus, can be solved by standard optimization solvers (e.g. CPLEX).

The outcome of the winner determination model is allocative efficient, as long as buyers and sellers reveal their valuations truthfully. The incentive to set bids according to the valuation is induced by an efficient *Pricing* mechanism. MACE implements a *k*-pricing scheme and as such determines prices for buyers and sellers on the basis of the difference between their bids. For instance, presume a buyer wants to buy a computation service for 5 and a seller wants to sell a computation service for at least 4. The difference between these bids is □ =1, i.e. □ is the surplus of this transaction and can be distributed among the participants. This schema can be applied to MACE and results in an approximate efficient outcome (Schnizler, Neumann et al. 2004).

The outcome of the mechanism, i.e. the allocation and the corresponding prices, is subsequently sent to the service providers and requesters as WS-Agreement.

5 POLICY-BASED AUTOMATED BIDDING

In this section, the reasoning behavior of the bidding agent is introduced, which automatically derives one or several bids based on system as well as market information. In essence, the generation is guided by policies. Policies are considered as declarative rules defined by the user to adapt and control the behavior of the system. Hence, policies are apt to embed the logic for bidding, which is in line with the business model of the service provider and requester, respectively.

To be effective, the bid generator obtains current system information from *(i)* the local resource manager of an application or computational nodes and *(ii)* the market mechanism. By means of policies this information is interpreted and appropriate actions (e.g. submission or manipulation of bids) are initiated. Figure 3 depicts the general bid generation process performed by the bidding agent.

In a first step, monitoring information, derived from the resource manager or the application, is explored to assess whether or not a bid should be conveyed to the market mechanism. This decision is based on the current and on the predicted utilization of the resource or on the expected demand of the application as well as on the preferences of the user formalized via bidding policies. In case demand or supply situation requires the submission of one or more bids, the concrete properties of the bids are determined by taking information from the market mechanism, as well as utilization, and user policies into account (second step). In a third step, the bid is expanded by means of a domain ontology that contains taxonomical and composition information. This step assures that different levels of abstraction in descriptions of bids do not lead to inefficiencies in the market. In addition, it reflects the fact that several Grid services can be offered and requested within one order. Finally, before forwarding to the market the bid is serialized using a WS-Agreement template to assure interoperability with the market mechanism. In the following, those four steps are described in more detail.

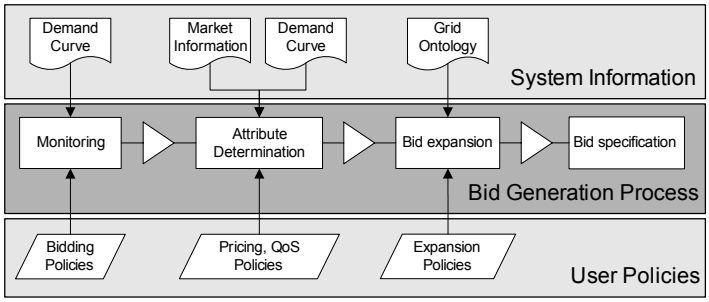

Figure 3: Bid Generation Process

5.1 First Step: Monitoring

The bid generation process is initiated by the decision that a bid needs to be generated. This decision is contingent upon the current and expected utilization of the resource or the demand of the application, respectively. The functions, which represent either the availability of the resource or the demand of an application are referred to as *load curves D(t)* in the following. Load curves are continuously obtained from the resource manager by means of prediction models. The constant *c* specifies the overall capacity that can be offered or requested. An example for such a load curve is shown in Figure 4.

Based on this load curve, as well as, on the bidding policies defined by the user a decision is made. For example, a simple bidding policy may specify that a bid must be send to the market in case the demand is predicted to exceed a certain threshold □ within the planning interval t_0 to t_1[2]. That means, a bid has to be submitted in case $D(t) > \square c$ holds at least for one t with $t_0 < t < t_1$. The policy specifying if a bid *b* has to be generated within a certain planning interval *p* based on threshold value □ as well as the overall capacity *c*:

issueBid(?b,?p) □ *PlanningInterval(?p), lowerBound(?p,?t1), upperBound(?p,?t2), Time(?t), load(?t,?dt), greaterThan(?t,?t1), greaterThan(?t2,?t),hasThreshold(?b,?v), hasCapacity(?b,?c), multiply(?v,?c,?e), greaterThan(?dt,?e)*

The length of the planning interval should be determined according to the minimal slot that can be requested from one single service provider or that can be provided to a single requester, respectively. Long-range planning intervals may thereby entail inaccuracies due to load changes during the interval. Thus, it is referred to mean values. For each interval this step will be executed only once. Moreover, machine learning approaches can be applied in order to adapting □ with respect to changing environments. However, a detailed description of these approaches goes beyond the scope of this paper.

To illustrate this approach consider a storage service provider which is exposed to the load curve depicted in Figure 4.

In doing so, we assume a planning interval of t_0=60s and t_1=80s as well as load levels of 300GB and 500GB within this interval, while assuming an overall capacity of c=800GB. Further, the bidding policy defines that a bid should be issued in case less than half of the overall capacity is used. Therefore, the threshold value is specified as θ=0.5. Since in the time frame between 60s and 70s the available storage exceeds $\theta c = 0.5 * 800GB = 400GB$ a bid has to be conveyed for the corresponding planning period.

5.2 Second Step: Attribute determination

After the bid generation has been triggered by the monitoring step the attribute values for a concrete bid have to be determined. Within each bid the type of Grid service that is required or offered, quality of service criteria that have to be fulfilled, a time interval in which the resource is offered/requested, and a range of prices which are acceptable (e.g. maximal or minimal price) has to be specified in a way that it is optimal with respect to the goals of the agent. These goals are influenced by three aspects: The user preferences reflected by the system policies, the system monitoring information, and the current market information. In the following we show how these three information sources can be integrated in order to derive attribute values for the individual attributes.

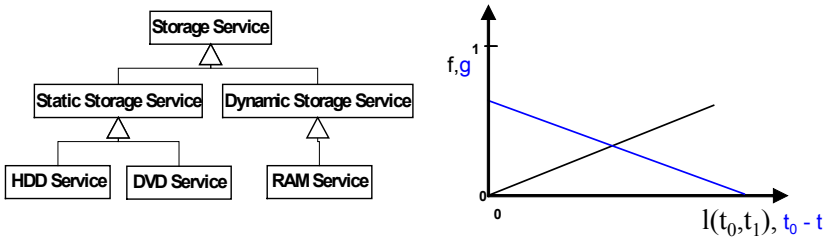

Figure 5: Storage Service Ontology and Eagerness Policies

Service Functionality: The service functionality determines which service is provided by the computational node, or requested by an application. This information is directly derived from the resource manager and can be modeled by referring to the corresponding concept of a Grid ontology, which is based on the Resource Specification Language (RSL) (Czajkowski, Foster et al. 2004), the Classified Advertisements (ClassAds) (Raman, Solomon et al. 2004) and domain-specific extensions as shown for the storage service example in the left panel of Figure 5. In a subsequent step, the relations between the different concepts in the ontology are exploited to improve the matchmaking of services in the market.

Price: The price for a Grid service is determined by considering (a) the load curve and the resultant eagerness factor, (b) the current price of similar services in the market and finally (c) the pricing policies of the user.

(a) In order to determine the load that is required from the application within the considered time frame [t_0, t_1] the equation $l(t_0,t_1) = \max_{t\in[t_0,t_1]}(D(t))$ is applied to ensure that the demand for that entire period can be fulfilled. For the provider, in contrast, the equation $l(t_0,t_1) = \min_{t\in[t_0,t_1]}(D(t))$ is used to avoid overstraining the resource.

The eagerness factor μ specifies how urgent suitable requesters or providers have to be found. Eagerness depends on the one hand on the load of the system in [t_0, t_1] and on the other hand on concession rate imposed by the remaining time span t_0 - t, where t represents the point in time when the bid is generated. Thus, the eagerness factor is determined by the equation $\mu = f(l(t_0,t_1)) + g(t_0 - t)$, where the functions f and g can be considered as eagerness policies, which assign to each load $D(t)$ and each time span t_0 - t a corresponding factor μ. The right panel of Figure 5 reflects common eagerness policies for the storage example, which rests on the intuition that the pressure to identify a corresponding request increases with growing load and decreases with shortening remaining time. Such policies are also formalized via SWRL-rules. The range of the eagerness factor is defined between zero and a positive rational number R. According to the policies in our running example this might lead to $f(300GB) = 0.3$ and $g(60s\text{-}0s)=0.6$. Consequently, we get an eagerness factor of $\mu=0.9$.

(b) The eagerness factor μ is now used to adapt the current market price m of a service with the same functionality and similar quality characteristics. This is achieved by means of the following equation, where p represents the proposed price: $p = m\,\mu$. Note that in case μ is greater than one, the price of the bid might surpass the current market price. In our running example this leads to 90% of the current market price.

(c) Assuring that the price does not adopt any undesired value, the provider might additionally specify pricing policies, which define the valid range [p_{low}, p_{high}] for the service prices. The simple pricing policies (b) and (c) are expressed via the following rules:

$price(?s,?p) \leftarrow Service(?s), minPrice(?s,?p_{min}), maxPrice(?s,?p_{max}), marketPrice(?s,?m), hasEagernessFactor(?\mu),$

multiply(?m,?μ,?p), greaterThan(?p,?p_{min}), smallerThan(?p,?p_{max})

price(?s,?p) □ *Service(?s), minPrice(?s,?p), marketPrice(?s,?m), hasEagernessFactor(?μ), multiply(?m,?μ,?e), smallerThan(?e,?p)*

price(?s,?p) □ *Service(?s), maxPrice(?s,?p), marketPrice(?s,?m), hasEagernessFactor(?μ), multiply(?m,?μ,?e), greaterThan(?e,?p)*

Quality of Service (QoS): The service offers and requests need to specify the guaranteed or requested QoS level, since this information is crucial for the subsequent agreement. However, QoS attributes are highly domain dependent and thus may vary from one service to another. Our notion of QoS includes attributes ranging from the capacity of a storage service to the response time or error rate of a computational service.

Partly information about quality requirements or quality guarantees can be automatically derived from the computing node or from the application, respectively. For example, as indicator for the capacity that can be offered or is required in the time interval $[t_0, t_1]$ the value $l(t_0, t_1)$ can be used. For this purpose we use the formula $cap = \square\, l(t_0, t_1)$, where the rate □ allows systematic over-/under-selling or over-/under-buying. In many scenarios □ is required; especially when contractual penalties are high (typically □ < 1) or working to capacity (typically □ > 1) is important. In case of a storage provider a rate of □ = 0.95 might be reasonable to reduce the risk of overselling. This means in the example we would offer storage capability of cap = □ $l(t_0, t_1) = 0.95 * 300GB = 285GB$.

If quality characteristics are not affected by the application or the resource, the QoS policies are directly used for the bid specification, by directly translating them into WS Policy documents. For instance, this could be policy which specifies that all communication with the service has to be encrypted.

Time Interval: Since the time interval $[t_0, t_1]$ represents the minimal period which can be sold to a single buyer or bought from a single provider the planning interval $[t_0, t_1]$ should also be the slot that is offered or requested in the bid. Moreover, the market mechanism that is used within the framework supports aggregation of quality criteria and time. This means, several time slots can be aggregated in order to reach an agreement in the market. Therefore, the issue of choosing slots that are too short is not relevant in this context. That mans, a storage request of 200 GB in the interval between 60s and 100s might be allocated to two storage offers: One in the interval [60s, 80s] and the other in the interval [80s, 100s], both with a capacity of 200GB.

5.3 Third Step: Bid expansion using background ontology

Having generated the initial bid which reflects the current system status, the aim of this section is to improve the probability for finding a suitable request in the market by expanding the bid in different directions. On the one hand, we analyze a Grid service taxonomy in order to introduce alternative offers that can also be used to describe the corresponding service. This is referred to as vertical bid expansion. On the other hand, we use equivalence relations explicitly modeled in an ontology to address requests that could also be fulfilled by a combination of different services. This approach is denoted by horizontal bid expansion.

5.3.1 Vertical bid expansion

Since offers and requests might be defined on different levels of abstraction pure syntactic matching often fails. Typically offers can be described very detailed since providers know exactly what kind of service they provide (e.g. Hard Disk Service), whereas customers usually define their requests on a rather abstract level (e.g. Storage Service). Consequently, in the market there will be no match between Hard Disk Service and Storage Service since the terms are compared only using strings. Therefore, we employ a Grid service ontology (as the example in Figure 5) that serves as background knowledge for adding additional XOR-related bids. By introducing these alternative bids according to Algorithm 1 we can overcome the different levels of abstraction while avoiding inappropriate matches.
Figure 5 shows a taxonomy for storage services which is used to exemplify how bids can be vertically expanded. Note that since we use formal ontologies the taxonomic relations either are modeled explicitly or they are derived by means of automatic classification done by a reasoner. According to Algorithm 1 in case of offers we introduce an additional XOR-related bid for each concept that is passed when traversing to the root of the hierarchy. For instance, a *HDD Service* offer is expanded by adding an offer for a *Static Storage Service* and an offer for a *Storage Service*. This is necessary since a HDD Service also fulfills the functionality of a Static Storage Service and a Storage Service and thus should be matched with a request for the latter. In case of a request we have to traverse to the leafs of the taxonomy since more specific services also fulfill the functionality of the more general services, e.g. a HDD Service meets the requirements for a Storage Service request. However, since this expansion could lead to a high number of alternative bids it might be necessary to limit the depth of such an expansion, which can be done by means of an *expansion policy*.

5.3.2 Horizontal bid expansion

Often a requested Grid service can not only be fulfilled by one single service, but also by a combination of different services. In order to reflect this, the Grid ontology allows expressing composite services and equivalence relations. As shown in Figure 6 a service is either an atomic or a composite service. A composite service is defined by the services it contains. Further, an equivalence relation can be defined between different services. This enables us to specify explicitly which services provide the same functionality. By means of the *equivalent_to* and *contains relations* new alternative bids can be introduced automatically to improve the probability or finding a suitable match in the market. For example, a mainframe computing service containing calculation as well as storage service can be defined by the following axioms:

Based on these definitions we are able to expand a request for a mainframe computing service by adding a XOR-related bundle request containing calculation as well as storage services. On the provider side we can expand a mainframe offer with a XOR-related bundle offer containing a storage and calculation service.

5.4 Bid Specification

Based on these results the final bid is formalized using WS Agreement as introduced in (Ludwig, Dan et al. 2004). For example, a storage service provider might specify the offer shown in Figure 7. Here the quality characteristics comprise a capacity of 285GB as well as a storage duration of 20s (planning time interval 60s-80s)[3]. Assuming that the average market price for such a service is 30 Euro the price in the bid adds up to 27 Euro since the urgency factor is 0.9. There is no vertical or horizontal bid expansion since storage service is already the top concept in the taxonomy, there are no *equivalent_to* relations and Storage Service is already an Atomic Concept in the ontology.

```
<wsag:AgreementOffer>
 <wsag:Terms>
  <wsag:All>
   <wsag:ServiceDescriptionTerm wsag:Name="readFile" wsag:ServiceName="StorageService">
    <job:arguments>/some/file/to/read</job:arguments>
   </wsag:ServiceDescriptionTerm>
   <wsag:All>
    <wsag:ServiceDescriptionTerm wsag:Name="storageSpace" wsag:ServiceName="StorageService">
     <job:realMemorySize>285</job:realMemorySize>
    </wsag:ServiceDescriptionTerm>
   </wsag:All>
   <wsag:GuaranteeTerm wsag:Name="Valuation">
    <wsag:ServiceScope>
     <wsag:ServiceName>StorageService</wsag:ServiceName>
    </wsag:ServiceScope>
    <wsag:BusinessValueList>
     <wsag:CustomBusinessValue>
      <mace:reservation>27</mace:reservation>
      <mace:timestart>60</mace:timestart>
      <mace:timeend>80</mace:timeend>
     </wsag:CustomBusinessValue>
    </wsag:BusinessValueList>
   </wsag:GuaranteeTerm>
 </wsag:Terms>
</wsag:AgreementOffer>
```

Figure 7: WS-Agreement Example

6 CONCLUSION

In recent times, the use of markets in resource allocation is increasingly proposed. In research, highly sophisticated market mechanisms are tailored for Grid. Although those algorithms work theoretically well, they require service requesters and providers to be aware of their demand and supply situation in explicit bids. Assuming that the service requesters and providers have this information certainly abstracts from the practical problems of retrieving demand and supply and valuing them accurately. If the bids are not reflecting the true demand and supply situation, the market price cannot also not reflect the correct situation. Thus, the market price looses its capacity to direct scarce resources to the bidders who value them most.

Accordingly, the establishment of markets for Grid needs supplementary bidder support. As the market mechanisms for Grid are very complex – accounting for the complementarities among the resources and the different QoS levels – the bidder needs adequate support. Since demand and supply fluctuations are very abrupt, bidding support needs to be automated. Current approaches of automated bidding are not applicable for Grid, as these works regard the price as only criterion. Approaches that go beyond the price are hitherto not available. This paper is unique by providing a policy-based bidding logic that can be straightforwardly automated. By means of policies the business models of the service requesters and providers can be incorporated. The policy-based approach is capable of integrating context information to the bidding process. In our example, we illustrate how the bid generator can submit several bids configuring more complex services. The bid generation process is currently quite simple by referring to observations and straightforward rules. In the future this process needs to be improved with respect to pricing (incorporating minimum probabilities of receiving the resources). Subsequently, the process will be implemented as proof-of-concept.

As such, the use of machine learning methods will be evaluated. Altogether, the policy-based approach bears the potential to automate the bidding process in Grid and contributes the missing piece that Grid markets find their way into practice.

REFERENCES

AuYoung, A., B. N. Chun, et al. (2004). Resource Allocation in Federated Distributed Computing Infrastructures. Proceedings of the 1st Workshop on Operating System and Architectural Support for the On-demand IT InfraStructure.

Baader, F., D. Calvanese, et al. (2003). The Description Logic Handbook: Theory Implemenation and Applications, Cambridge University Press.

Czajkowski, K., I. Foster, et al. (2004). Resource and Service Management. The Grid 2 - Blueprint for a New Computing Infrastructure, Elsevier. **2:** 259–283.

Dumas, M., L. Aldred, et al. (2005). "Probabilistic Automated Bidding in Multiple Auctions." Electronic Commerce Research **5**(1): 25 - 49.

Foster, I. and C. Kesselman (2004). The Grid 2. San Francisco, Morgan Kaufmann.

Huberman, B. A. (1988). The Ecology of Computation. Amsterdam, North-Holland.

Irwin, D. E., L. E. Grit, et al. (2004). "Balancing Risk and Reward in a Market-based Task Service." Working Paper.

Kee, Y.-S., H. Casanova, et al. (2004). Realistic Modeling and Synthesis of Resources for Computational Grids. ACM Conference on High Performance Computing and Networking.

Lai, K. (2005). "Markets are Dead, Long Live Markets." Working Paper.

LSF (2005). LSF, http://www.platform.com.

Ludwig, H., A. Dan, et al. (2004). Cremona: An Architecture and Library for Creation and Monitoring of WS-Agreements. 2nd International Conference on Service Oriented Computing (ICSOC 2004).

Motik, B., U. Sattler, et al. (2005). "Query Answering for OWL-DL with Rules." Journal of Web Semantics: Science, Services and Agents on the World Wide Web **3**: 41-60.

Raman, R., M. Solomon, et al. (2004). The ClassAds Language. Grid Resource Management -- State of the Art and Future Trends. J. Nabrzyski, J. M. Schopf and J. Weglarz, Kluwer Academic Publishers**:** 255-270.

Schnizler, B., D. Neumann, et al. (2004). Resource Allocation in Computational Grids - A Market Engineering Approach. WeB 2004, Washington.

Shneidman, J., C. Ng, et al. (2005). Why Markets Could (But Don't Currently) Solve Resource Allocation Problems in Systems. 10th USENIX Workshop on Hot Topics in Operating Systems (HotOS-X), Santa Fe, NM.

(Footnotes)

[1] Globus Toolkit is a reference implementation of the Open Grid Services Architecture (OGSA) which is the de-facto Grid architecture. See http://www.globus.org/ for details.

[2] Note that for a service requester the threshold □ will usually be zero. However, for service provider it might be reasonable to submit a bid only in case the workload falls below a certain level.

[3] The valuation prices and time restrictions are not part of the standard WS-Agreement specification. WS-Agreement permits, however, the extension of the specification via domain specific schemas. Thus, the prices are included as as domain specific schemas as a subset of the CustomBusinessValue tag using a separate XML-schema.