Technology Management, Economics and Policy Papers

SEOUL NATIONAL UNIVERSITY
College of Engineering
Technology Management, Economics and Policy Program

Enabling Open Cloud Markets Through WS -Agreement Extensions

Marcel Risch, Jörn Altmann

TEMEP Discussion Paper No. 2010:20

서 울 대 학 교
공 과 대 학, 기술경영경제정책 대학원과정
151-742 서울시 관악구 관악로 599

Technology Management, Economics and Policy Program
College of Engineering, Seoul National University
599 Gwanak-Ro, Gwanak-Gu, Seoul 151-742, South-Korea
Phone: ++82-2-880-9140, Fax: ++82-2-880-8389

Technology Management, Economics and Policy Program (TEMEP) is a graduate program at the Seoul National University. It includes both M.Sc. and Ph.D. programs which are integrated into the graduate program at the College of Engineering.

TEMEP Discussion Papers is intended to serve as an outlet for publishing research about theoretical, methodological and applied aspects of industrial economics, especially those related to the institute's areas of specialization, namely management of technology, information and telecommunication, health and energy industries, as well as development economics infrastructure. In particular submission of studies analyzing current technology and industry related issues and discussion of their implications and possible alternative policies are welcome. The objective is to gain insights into important policy issues and acquiring a balanced viewpoint of policymaking, technology management and economics which enable us to identify the problems in the industries accurately and to come up with optimal and effective guidelines. Another important aim with the series is to facilitate communication with external research institutes, individual researchers and policy makers.

Research disseminated by TEMEP may include views on information and telecommunication, technology management, energy, health and development economics policy, but the institute itself takes no institutional policy positions. When appropriate the policy views of the institute are disseminated in separate policy briefs. Thus, any opinions expressed in TEMEP Discussion Papers are those of the author(s) and not necessarily the institutes.

Editor: Almas Heshmati
Professor of Economics

Technology Management, Economics and Policy Program
College of Engineering
#37-306, Seoul National University
San 56-1, Shilim-dong, Kwanak-gu
Seoul 151-742 Korea

Phone: 0082-(0)2-880-5845
Fax: 0082-(0)2-886-8220
E-mail: heshmati@snu.ac.kr

Enabling Open Cloud Markets Through WS-Agreement Extensions

Marcel Risch and Jörn Altmann

Seoul National University
San 56-1, Sillim-Dong, Gwanak-Gu,
Seoul, 151-742, South-Korea

marcel.risch@temep.snu.ac.kr,
jorn.altmann@acm.org

October 2009

Abstract: Research into computing resource markets has mainly considered the question of which market mechanisms provide a fair resource allocation. However, while developing such markets, the definition of the unit of trade (i.e. the definition of resource) has not been given much attention. In this paper, we analyze the requirements for tradable resource goods. Based on the results, we suggest a detailed goods definition, which is easy to understand, can be used with many market mechanisms, and addresses the needs of a Cloud resource market. The goods definition captures the complete system resource, including hardware specifications, software specifications, the terms of use, and a pricing function. To demonstrate the usefulness of such a standardized goods definition, we demonstrate its application in the form of a WS-Agreement template for a number of market mechanisms for commodity system resources.

Keywords: Grid economics, unit of trade, system resource trading, Cloud computing market, system virtualization, contract templates, WS-agreements, service level contract, Cloud economics.

JEL Classification: D80, D83, L14, L86, M15, M21

1 Introduction

There have been many proposals for Grid markets, such as Spawn [1], Tycoon [2], GRACE [3], GridEcon [4], and MACE [5]. The main goal of these projects was to develop a market mechanism, which would allow for an efficient and fair allocation of computing resources. To achieve this goal, some projects have decided to focus on the most important aspect of a computing resource: its computational power.

These simplifications were acceptable since some applications predominantly require computing power. Furthermore, since designing a market mechanism for complex goods is difficult, initial simplifications had to be made for implementing a proof-of-concept. However, since the development of computing resource markets has progressed and a number of commercial offerings are now available, these simplifications must be removed and the tradable good must be described in its entirety to ensure that traders in a computing resource market can describe their offers and demands accurately.

In this paper, we propose a detailed definition of a general scheme for defining computing resources (i.e. units of trade) that consists not only of the hardware but also of software and the terms of use. A complete resource contract, which can also be called Service Level Agreement, contains the definition of system resources, the software purchased with it (if applicable), the pricing function, and the guarantees given by the provider. Such a contract would capture all aspects that are relevant to trading of Cloud resources.

Traditionally, the guarantees given by the provider and the duties of both providers and buyers have been captured by Service Level Agreements (SLAs). While many works on computing resource markets assume the existence of Service Level Agreements [7] [8], there is very little work on the exact definition of a SLA for resources. The best attempt has been made by the OGF GRAAP working group [9], which provided the WS-Agreement scheme [10]. However, this scheme is not aimed at system resources in particular but rather at general services. We will demonstrate how WS-Agreement can be used as a basis for developing a Computing Resource Definition Language (CRDL), which can be used in the setting of computing resource markets. Finally, we show how this new WS-Agreement extension can be used in a number of different market mechanisms and show that such a resource description will simplify resource trading.

2 State of the Art

2.1 WS-Agreement

WS-Agreement was developed to allow service consumers and service providers to form contracts which specify the service details, the guarantees, the obligations, and the penalties for each party concerned. This is, of course, exactly what is needed in Cloud computing markets, since Cloud computing resources are services.

WS-Agreement comprises three major sections: Name, Context, and Terms. The Name section contains the optional name of the agreement and a unique ID. In the Context section, information about the service provider and consumer are noted, as well as the expiration time of the agreement. The Terms section contains information

about the service terms and the guarantees. The service terms describe the service in as much detail as possible, while the guarantees describe which minimum performance the service will provide.

In general, WS-Agreement can be used to describe services of any kind. At the same time, it also implements a negotiation model for negotiating an agreement. This versatility should make WS-Agreement a very useful tool in open Cloud markets. However, the fact that WS-Agreement is not intended to be specific to any type of market makes it very difficult to use for computing services. We therefore will use the basic structure of WS-Agreement to develop a Computing Resource Definition Language (CRDL), which can be used in Cloud computing markets.

2.2 Existing Commercial Cloud Offers

In recent years, a large number of commercial Cloud providers have entered the utility computing market, using virtualization. Now, there are a number of different types of services which are sold under the label of "Cloud Computing". On the one hand, there are resource providers, such as Amazon (e.g. EC2 [16]) and Tsunamic Technologies [17], who provide computing resources. On the other hand, there are providers, who not only sell their own computing resources but also their own software services, such as Google Apps [18] and Salesforce.com [19]. Furthermore, there are companies that attempt to run a mixed approach, i.e. they allow users to create their own software services but, at the same time, the company offers various support services (i.e. platform services) to its customers. An example of such an approach is the Sun N1 Grid [20].

Looking at these resource providers, it should be noted that none of them use WS-Agreement to describe the SLAs. Instead, some of these providers use their legal staff to draft the SLA in human-readable format [42]. Others, such as Sun and Tsunamic Technologies, do not even provide SLAs publicly. The fact that these providers do not use WS-Agreement for their SLAs indicates that WS-Agreement still has some major shortcomings.

2.3 Open Cloud Market Enablers

In addition to the introduction of virtualization in data centers, several companies (e.g. Enomaly [38] or Fluid Operations [45]) now offer platforms to integrate in-house resources with externally purchased Cloud resources. These products not only allow users to turn their data center into a Cloud but also allow users to act as providers for resource services. Using such software, data center operators could easily be encouraged to participate in an open Cloud market, since the integration of internal and external resources is simplified.

A company that follows this idea is Zimory [39]. Its product turns a regular data center into an intra-company Cloud, allowing an efficient use of resources. In addition, Zimory also allows its costumers to sell spare capacity via its own marketplace. During times of high demand, of course, resources can then be purchased via the Zimory Cloud.

At this time, none of these companies seem to offer the capability to use any kind of SLAs with their services. However, if an open Cloud market were to be established, provisions for legally binding contracts would have to be made. Since WS-Agreement

already uses a very clear structure, this template could be adapted to be more suitable for Cloud resources.

2.4 Computing Resource Markets Research

The research into system resource markets can be divided into two groups, when looking at their description of tradable goods. The first group does not define goods at all, while the second group focuses on one aspect of a computing resource only.

The first group consists to a large extend of early Grid market designs. Examples for these early designs are GRACE and designs by Buyya [11]. In these early designs, the analysis of Grid market entities and the architecture of such a market have been analyzed. However, the good "computing resource" has not been defined. Since the tradable good has not been considered, the question of how the contractual obligations can be defined has also not been addressed.

The second group of Grid market research has simplified the computing resource good. The MACE exchange takes a small step away from the initial market architectures [5]. The authors recognize the importance of developing a definition for the tradable good. However, they abstract computing resources into services that can be traded; the actual definition has never been supplied.

Another approach, which was taken by several research groups, was the focus on computing power, either in the form of Java OPerations (JOPs) within the Popcorn market [6], or in the form of CPU slices within the Spawn market [1]. These goods are very restrictive, since they require detailed knowledge about the factors (e.g. application requirements, the compiler vendor, the instruction sets of the CPU) that influence the amount of computing power needed.

Lastly, there is the Tycoon market [2], which was developed before virtualization tools (e.g. Xen [12]) became widely used. The initial stages worked with basic computing cycles and it was planned to extend this market by making use of virtualization. However, it seems that the effort has been discontinued.

Overall, much of the resource market research has worked with either simplified definitions of the tradable good or without defining the good at all. Therefore, since Grid research did not provide any foundation for defining a computing resource good, we will use WS-Agreement to develop such a definition. WS-Agreement was chosen for its flexibility: it will be used as a basis for developing a Computing Resource Definition Language, keeping the structure of WS-Agreement but expanding it to describe resources in detail. The usefulness of WS-Agreement for defining extensions has been shown quite frequently [48-50].

3 Extending WS-Agreement

3.1 Diversity of Goods

In an open market environment, diverse computing resources have to be described in a common format so that customers can determine the differences between various resource offers. This diversity may seem daunting at first but it should be remembered that trading diverse resources is already possible in practice for other goods. Looking at the Chicago Mercantile Exchange (CME) [23], we can see that even diverse products

such as live cattle can be traded. Since live animals are extremely diverse, the trading contracts are very detailed, including penalties and obligations, as the following example shows:

"A par delivery unit is 40,000 pounds of USDA estimated Yield Grade 3, 55% Choice, 45% Select quality grade live steers, averaging between 1,100 pounds and 1,425 pounds with no individual steer weighing more than 100 pounds above or below the average weight for the unit. No individual animal weighing less than 1,050 pounds or more than 1,475 pounds shall be deliverable. […] Steers weighing from 100 to 200 pounds over or under the average weight of the steers in the delivery unit shall be deliverable at a discount of 3¢ per pound, provided that no individual animal weighing less than 1,050 or more than 1,475 pounds shall be deliverable." (pg. 3, [24])

This principle of a complete contract should be adopted for infrastructure markets so that all parties (buyers and sellers) involved in the trade can easily understand the properties of the traded good and have a common reference to the traded resource.

3.2 Composition of the Service Level Contract

The benefits of using WS-Agreement to compose Service Level Contracts lies in the fact that WS-Agreement already comprises all categories necessary for describing a computing resource definition. Therefore, all that has to be done is to extend WS-Agreement in such a way that it becomes more specific. All excerpts from WS-Agreement in this chapter have been taken from [41]. The basic structure of WS-Agreement consists of a Name, an AgreementContext, and a Terms section. In a market environment, in which the resource definitions are managed centrally, the agreement name will be specified in advance. The agreement ID will only be created when a buyer and seller decide to trade resources.

3.2.1 Agreement Context

The AgreementContext section contains information about the service provider and the agreement initiators and respondents. The full specification of the AgreementContext section is shown below (note, all extensions to the WS-Agreement that we suggest are indicated in italic throughout the remainder of the paper):

```
<wsag:Context xs:anyAttribute>
  <wsag:AgreementInitiator>xs:anyType</wsag:AgreementInitiator> ?
  <wsag:AgreementResponder>xs:anyType</wsag:AgreementResponder> ?
  <wsag:ServiceConsumer>wsag:AgreementRoleType</wsag:ServiceConsumer>
  <wsag:ServiceProvider>wsag:AgreementRoleType</wsag:ServiceProvider>
  <wsag:StartingTime>xs:DateTime</wsag:StartingTime>
  <wsag:EndingTime>xs:DateTime</wsag:EndingTime>
  <wsag:PurchaseTime>xs:DateTime</wsag:PurchaseTime>
  <wsag:ExpirationTime>xs:DateTime</wsag:ExpirationTime>
  <wsag:TemplateId>xs:string</wsag:TemplateId> ?
  <wsag:TemplateName>xs:string</wsag:TemplateName> ?
  <xs:any/> *
</wsag:Context>
```

The AgreementInitiator and AgreementResponder sections will have to be filled according to the market mechanism, but can be left blank if they are not needed. The Service Provider section, as well as the newly created ServiceConsumer section, will be filled with information about the service provider and the service consumer, respectively. Both fields are mandatory.

The StartingTime section allows for future delivery of the service. Since even spot market sales have a lag time, which allows the provider to set up the resources, this starting time is vital to ensure that the agreement covers the relevant time schemes. We also introduced an EndingTime section, which denotes the time at which the resources revert to their owner. Both these fields are obligatory. The PurchaseTime section describes the time of purchase of the SLA. This is necessary for some pricing function as explained in section 3.2.4.

The ExpirationTime section will be filled in according to the expiration time of the contract. This time must be at least the same time as the ending time to ensure that the contract is valid for the entire duration of the resource usage. The TemplateID and TemplateName sections will be filled as needed.

3.2.2 Terms

The Terms section defines both the service that is traded, as well as the guarantees that the provider is willing to make. Furthermore, this section can contain a description of the penalties as well as a description of the restrictions. The Terms section contains elements for the service description, service reference, service properties, and the guarantee terms.

While this structure is very general, for computing resources however, the ServiceDescriptionTerm section can be used to describe the resource in its entirety. The extension to the WS-Agreement contains the information about the CPU, main memory, hard disk, and the network. A detailed description of a resource is shown below on the left.

```
<xs:complexType name="ServiceDescriptionTerm">
 <xs:sequence>
    <xs:complexType name="Resource">
       <xs:sequence>
          <wsag:ExactlyOne>
             <xs:element name="Processor"
             type="wsag:ProcessorType"/>
             <xs:element name="MainMemory"
             type="wsag:MainMemoryType"/>
             <xs:element name="HDD"
             type="wsag:HDDType"/>
             <xs:element name="Network"
             type="wsag:NetworkType"/>
             <xs:element name="BusSpeed"
             type="wsag:BusSpeedType"/>
             <xs:element
             name="NumberOfResources"
             type="xs:integer"/>
          </wsag:ExactlyOne>
       </xs:sequence>
    </xs:complexType>
 </xs:sequence>
</xs:complexType>
```

```
<xs:element name="Processor">
 <xs:complexType>
    <xs:sequence>
     <wsag:ExactlyOne>
       <xs:element name="Brand"
       type="xs:string"/>
       <xs:element name="ProcessorName"
       type="xs:string"/>
       <xs:element name="NumberOfCores"
       type="xs:integer"/>
       <xs:element name="ProcessorSpeedGHz"
       type="xs:decimal"/>
       <xs:element name="L1CacheMB"
       type="xs:decimal"/>
       <xs:element name="L2CacheMB"
       type="xs:decimal"/>
       <xs:element name="FSBMHz"
       type="xs:decimal"/>
    </wsag:ExactlyOne>
    </xs:sequence>
  </xs:complexType>
 </xs:element>
```

Figure 1. ServiceDescriptionTerm and processor definition.

Most of these complex types will have to be expanded. On the right-hand side of Figure 1, we demonstrate how the processor can be described in detail. Similarly, the main memory will have to be described in more detail, as shown on the left side of Figure 2. The right side of Figure 2 shows how the hard disk can be specified in detail.

```
<xs:element name="MainMemory">
 <xs:complexType>
    <xs:sequence>
       <wsag:ExactlyOne>
          <xs:element name="Type"
          type="xs:string"/>
          <xs:element name="SizeGB"
          type="xs:decimal"/>
          <xs:element name="MemoryClockMHz"
          type="xs:integer"/>
          <xs:element name="CycleTimeNs"
          type="xs:integer"/>
          <xs:element name="IOBusClockMHz"
          type="xs:integer"/>
          <xs:element
          name="DataTransferMBitPerSecond"
          type="xs:integer"/>
       </wsag:ExactlyOne>
    </xs:sequence>
 </xs:complexType>
</xs:element>
```

```
<xs:element name="HDD">
 <xs:complexType>
    <xs:sequence>
       <wsag:ExactlyOne>
       <xs:element name="SizeGB"
       type="xs:decimal"/>
       <xs:element name="RPM"
       type="xs:integer"/>
       <xs:element name="SeekTimeMS"
       type="xs:decimal"/>
       <xs:element
       name="DataTransferRateMbitPerSec"
       type="xs:decimal"/>
       <xs:element name="CacheSizeMB"
       type="xs:decimal"/>
       </wsag:ExactlyOne>
    </xs:sequence>
 </xs:complexType>
</xs:element>
```

Figure 2. Description of memory and harddisk.

Next, we show how the network access and the software parameters can be described. This is shown in Figure 3 below.

```
<xs:element name="Network">
 <xs:complexType>
    <xs:sequence>
       <wsag:ExactlyOne>
       <xs:element
       name="MaximumTransmissionSpeedMbitPe
       rSec" type="xs:decimal"/>
       <xs:element name="CacheSizeMB"
       type="xs:decimal"/>
       </wsag:ExactlyOne>
    </xs:sequence>
 </xs:complexType>
</xs:element>
```

```
<xs:element name="Software">
 <xs:complexType>
    <xs:sequence>
       <wsag:ExactlyOne>
       <xs:element
       name="VirtualizationSoftware"
       type="xs:string"/>
       <xs:element name="OperatingSystems"
       type="xs:string"/>
       <xs:element name="StaticIP"
       type="xs:boolean"/>
       </wsag:ExactlyOne>
    </xs:sequence>
 </xs:complexType>
</xs:element>
```

Figure 3. Description of network and software.

This software section describes whether a virtualization tool was used to create the resource, and if so, which types of software containers can be run on the resource. This allows the user to determine whether this resource is capable of running software containers already owned by the user, or if new containers have to be created. Furthermore, if the provider desires, this category can also describe which operating systems can run on the resource or is pre-installed. This is especially important if virtualization tools are used, since these tools place some restrictions on the operating systems that can be deployed. In addition, this category can describe whether static or dynamic IP addresses are used.

3.2.3 Guarantee Terms

The GuaranteeTerms section describes the guarantees the service provider will have to give. In the case of computing resources, the resource description given in the ServiceDescription section is a minimum performance that must be provided. To see how the WS-Agreement GuaranteeTerms can be extended, we will have to look at this section in more detail. The GuaranteeTerms consists of the ServiceScope section, the QualifyingCondition section, the ServiceLevelObjective section and the BusinessValueList section.

The ServiceScope describes to which element of the service term the guarantee applies (e.g. for software services, one element could be the response time). In the case of computing resources, the guarantees cover all parameters of the computing resource as a whole. The QualifyingCondition section describes when a guarantee starts to apply. In the case of computing resources, this is the moment when the user is given control of the resource. The ServiceLevelObjectives section describes when a guarantee is considered to be met. In the case of computing resources, this is the case if the resource is available for the specified amount of time and if all parameters in the description are met, e.g. the processor speed is correct and memory size is correct.

The BusinessValueList section describes penalties and rewards. It consists of the Importance, Penalty, Reward, Preference and CustomBusinesValue sections. The Importance section describes the importance of a given business value. This can be filled in, if the providers or consumers so desire.

The Rewards section describes the reward that could be incurred for meeting an objective. In the case of computing resources, this could be the price that has to be paid by the consumer for using the specified resource.

```
<wsag:Reward>
 <xs:element name="Price">
    <xs:complexType>
       <xs:sequence>
          <xs:element name="One_Time_Fee"
          type="xs:decimal"/>
          <xs:element name="CPU_HR"
          type="xs:decimal"/>
          <xs:element name="Upload"
          type="xs:decimal"/>
          <xs:element name="Download"
          type="xs:decimal"/>
          <xs:element name="Storage"
          type="xs:decimal"/>
       </xs:sequence>
    </xs:complexType>
 </xs:element>
</wsag:Reward>
```

```
<wsag:Preference>
 <xs:element name="PriceValidity">
    <xs:complexType>
       <xs:sequence>
          <xs:element name="BusinessValueListReference"
          type="wsag:BusinessValueListID"/>
          <xs:element name="CPU_HR_Lower_Usage_Bound" type="xs:decimal"/>
          <xs:element name="CPU_HR_Upper_Usage_Bound" type="xs:decimal"/>
          <xs:element name="CPU_HR_Min_Usage" type="xs:decimal"/>
          <xs:element name="Upload_Lower_Usage_Bound" type="xs:decimal"/>
          <xs:element name="Upload_Upper_Usage_Bound" type="xs:decimal"/>
          <xs:element name="Upload_Min_Usage" type="xs:decimal"/>
          <xs:element name="Download_Lower_Usage_Bound" type="xs:decimal"/>
          <xs:element name="Download_Upper_Usage_Bound" type="xs:decimal"/>
          <xs:element name="Download_Min_Usage" type="xs:decimal"/>
          <xs:element name="Storage_Lower_Usage_Bound" type="xs:decimal"/>
          <xs:element name="Storage_Upper_Usage_Bound" type="xs:decimal"/>
          <xs:element name="Storage_Min_Usage" type="xs:decimal"/>
          <wsag:StartingTime>xs:DateTime</wsag:StartingTime>
          <wsag:EndingTime>xs:DateTime</wsag:EndingTime>
          <xs:element name="Min_Usage_Time_Minutes" type="xs:decimal"/>
       </xs:sequence>
    </xs:complexType>
 </xs:element>
</wsag:Preference>
```

Figure 4. Reward and Preference sections.

The Preference section is used in cases where multiple business values apply to the service at the same time. In such a case, the preference indicates which business value should be used. In the case of computing resources, this section expresses pricing and multiple pricing models can be ranked according to their preference. This allows for different pricing models to be applied to different usage and quality levels. Note, if multiple prices apply for the same usage, the consumer will have to pay all fees.

The Penalty section describes the obligations of consumers and providers. In this case, consumers and providers can be held accountable, if certain conditions are not met. In the case of computing resources, these conditions can be quite diverse, ranging from limiting usage of hardware to excluding certain hardware infrastructures entirely from some customers. The definition of the Penalty section follows the one of the Reward section and uses the structure of the Preference section. If a Reward section and a Penalty section exist with the service level contract, two BusinessValueList sections have to be filled out.

The CustomBusinessValues section can be used to describe usage restrictions and responsibilities of the consumer. These are vital to the contract, since many aspects of the resource usage can be affected by these rules.

4 The System Resource Contract in Different Market Mechanisms

In this section, we will demonstrate that the proposed extension of WS-Agreement can be used in conjunction with many market mechanisms. In particular, we will focus on posted price, bargaining, and single auctions.

4.1 Posted Price

In a posted price market, providers are free to create any resource types, which they deem to be tradable. These freely created resource types can also come with highly individualized Terms of Use. For the Computing Resource Definition Language, this means that the Terms section would, in all likelihood, be different for different providers. However, in order to achieve understandability, the Terms would link to an external reference, where the Terms are written in natural language and can be seen by consumers.

In a posted price market environment, the offers could be centrally listed and potential customers can then search these offers based on their unit-of-trade requirements. Such matching procedures have been proposed in [46]. Using the CRDL, this search procedure will be simplified, since the vital parameters can be easily compared.

4.2 Negotiation Environment

In a negotiation process, a provider and consumer can negotiate the different categories of the computing resource contract individually. Should the need arise, both parties can even negotiate the individual aspects of each of the four contract categories. In this case, the contract template helps the consumer and the provider to follow a certain structure for their negotiation process.

Therefore, the CRDL does not hinder the bargaining partners to negotiate each aspect of a resource. Instead it simplifies the procedure if both parties can agree on standard components. In fact, any mix of standard and individualized components can be combined to form a new contract. Further simplifications are achieved by the fact that WS-Agreement already includes some tools for negotiations which can be used by the trading parties. Furthermore, the flexible pricing section allows both parties to define different prices for individual components. This will ensure that the negotiating parties can be certain that their pricing requirements are met.

4.3 Single Auction

In a single auction, the provider simply posts the service level contract without setting the price of the good. The resource consumer searches for suitable resources being currently auctioned and bids for resources using his bidding strategy. Depending on the auction, the consumer bids for the good and, if he wins, gives the purchase price to the auctioneer. The final service level agreement comprises the service level contract filled with the price associated with the winning bid of the consumer.

The advantage of using CRDL contracts lies in the easy-to-understand description of the traded good. The consumer can easily determine which capabilities each resource

has and what the provider is willing to guarantee. Furthermore, since the good is fully described and perhaps only the price is missing, the auction procedure does not need to be adapted to be able to work with other goods of this type of good. Therefore, any existing auction market can work with CRDL. The bid would be a simple value pair <contract ID, price>.

4.4 Discussion of the Computing Resource Definition Language

All market mechanisms use the same template to define the tradable good, i.e. the computing resource. While some market mechanisms, such as single auctions, require such a stringent structure, others are more lenient. Since WS-Agreement is used as a basis, traders will easily understand the structure. Furthermore, if the restrictions, penalties and responsibilities are defined in legal terms, the existing template can then be taken to court.

The comparability requirement of different Computing Resource Contracts is provided not only through a standard format, but also through the same structure of all contracts. This means that the important aspects of each contract can be easily found and can then be compared. However, this requires additional effort in standardizing SLAs, which has been started within the SLA framework of OGF [10].

The flexibility of CRDL is given by the fact that it can represent a large number of different resources types. This structure of a contract allows for individual parts to be, to some extent, standardized. The individual parts can still be combined in many ways to form contracts.

Since CRDL allows a wide array of resources to be described and resource descriptions with CRDL are understandable by all parties, comparisons of units of trades will be simple. Therefore, it is likely that the resource descriptions with CRDL will be accepted by a large number of users.

5 Conclusion and Future Work

In this paper, we have introduced the Computing Resource Definition Language (CRDL), an extension to WS-Agreement. It is aimed at capturing the complexity of system resources in a single descriptor, which can be traded in many market environments. The usefulness of this extension has been discussed by showing its application to different market mechanisms. The main goal was to enhance the usefulness of WS-Agreement for markets where many different market mechanisms can exist. In particular, we showed how CRDL can be applied to utility computing markets for virtualized goods.

References

1. Waldspurger, C.A., Hogg, T., Huberman, B.A., Kephart, J.O., Stornetta, W.S. "Spawn: A Distributed Computational Economy," IEEE Transactions on Software Engineering, vol. 18, no. 2, pp. 103-117, Feb., 1992.

2. Lai, K., Rasmusson, L., Adar, E., Zhang, L., and Huberman, B. A. Tycoon: An implementation of a distributed, market-based resource allocation system. Multiagent Grid Syst. 1, 3 (Aug. 2005), pp. 169-182.
3. Buyya R, Abramson D, Giddy J. An economy grid architecture for service-oriented grid computing. 10th IEEE International Heterogeneous Computing Workshop (HCW 2001), San Francisco, CA, April 2001. IEEE Computer Society Press: Los Alamitos, CA, 2001.
4. Altmann, J., Courcoubetis, C., Dramitinos, M., Stamoulis, G.D., Rayna, T., Risch M., Bannink, C., "A Market Place for Computing Resources," GECON 2008, Workshop on Grid Economics and Business Models, Springer LNCS, Las Palmas, Spain, August 2008.
5. Schnizler, B., Neumann, D., Veit, D., Weinhardt, C. Trading Grid Services - A Multi-attribute Combinatorial Approach. European Journal of Operational Research, 187(3). pp. 943-961 (2008).
6. Regev, O. and Nisan, N. 1998. The POPCORN market—an online market for computational resources. In Proceedings of the First international Conference on information and Computation Economies (Charleston, South Carolina, United States, October 25 - 28, 1998). ICE '98. ACM, New York, NY, 148-157.
7. Macias, M., Smith, G., Rana, O.F., Guitart, J., Torres, J. Enforcing Service Level Agreements using an Economically Enhanced Resource Manager. In: Workshop on Economic Models and Algorithms for Grid Systems (EMAGS 2007), Texas, USA (2007).
8. Sahai, A., Graupner, S., Machiraju, V., and Moorsel, A. v. 2003. Specifying and Monitoring Guarantees in Commercial Grids through SLA. In Proceedings of the 3st international Symposium on Cluster Computing and the Grid (May 12 - 15, 2003). CCGRID. IEEE Computer Society, Washington, DC, 292.
9. The Open Grid Forum (OGF), http://www.ogf.org/, 2008.
10. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu. M. Web Services Agreement Specification (WS-Agreement). GWD-R (Proposed Recommendation), Open Grid Forum, 2007.
11. Rajkumar Buyya, Sudharshan Vazhkudai, "Compute Power Market: Towards a Market-Oriented Grid," ccgrid,pp.574, First IEEE International Symposium on Cluster Computing and the Grid (CCGrid'01), 2001.
12. XenSource, Inc., http://xen.org/, 2008.
13. Altmann, J., Rupp, B., Varaiya, P., "Effects of Pricing on Internet User Behavior," NetNomics, vol.3, no.1, June 2000.
14. Walrand, J., He, L., "Dynamic Provisioning of service Level Agreements between Interconnected Networks," Allerton, Allerton, USA, 2002.
15. Shu, J., Varaiya P., "Pricing Network Services," Infocom 2003, vol.2, p.1221-1230, April 2003.
16. Amazon Elastic Compute Cloud (Amazon EC2), http://aws.amazon.com/ec2/, 2009.
17. Tsunamic Technologies Inc., http://www.clusterondemand.com/, 2008.
18. Google Apps, http://www.google.com/apps/, March 2009.
19. Salesforce.com, http://www.salesforce.com, March 2009.
20. Sun Grid, http://www.sun.com/service/sungrid/index.jsp, 2008.
21. Amazon EC2 Instance Types, http://aws.amazon.com/ec2/instance-types/, 2008.
22. McKendrick, J. "2009 share survey: Total enterprise virtualization", 2009.
23. Chicago Mercantile Exchange, http://www.cme.com/, 2008.

24. CME Rulebook, http://www.cmegroup.com/cmegroup/rulebook/CME/II/100/101/101.pdf, 2008.
25. Sun Grid International Access, http://www.sun.com/service/sungrid/whatsnew.jsp#int_avail, 2008.
26. Ouelhadj, D. Garibaldi, J.M., MacLaren, J., Sakellariou, R., Krishnakumar, K.: A Multi-agent Infrastructure and a Service Level Agreement Negotiation Protocol for Robust Scheduling in Grid Computing. EGC 2005: 651-660.
27. Pichot, A., Wieder, P., Waldrich, O., Ziegler, W. Dynamic SLA Negotiation based on WS-Agreement. CoreGrid, technical report, TR-0082, June-24, 2007.
28. Hasselmeyer, P.; Qu, C.; Koller, B.; Schubert, L.; Wieder, P. Towards Autonomous Brokered SLA Negotiation. Exploiting the Knowledge Economy: Issues, Applications and Case Studies, Volume 3 Information and Communication Technologies and the Knowledge Economy, P. Cunningham and M. Cunningham (eds.), IOS Press, Amsterdam, 2006, ISBN 978-1-58603-682-9, pp. 44-51.
29. Green, L., Mirchandani, V., Cergol, I., and Verchere, D. 2007. Design of a dynamic SLA negotiation protocol for grids. In Proceedings of the First international Conference on Networks For Grid Applications (Lyon, France, October 17 - 19, 2007). ICST (Institute for Computer Sciences Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, 1-8.
30. Quan, D.M., Kao, O. "SLA negotiation protocol for Grid-based workflows", Proceedings of the International Conference on High Performance Computing and Communications (HPPC-05), LNCS 3726, pp. 505-510, 2005.
31. MacLaren, J. , Sakellariou, R., Garibaldi, J., Ouelhadj, D. Towards Service Level Agreement Based Scheduling on the Grid. In the proceedings of the Workshop on Planning and Scheduling for Web and Grid Services, art of the 14th International Conference on Automated Planning & Scheduling (ICAPS 2004), Whistler, British Columbia, Canada, June 3-7, 2004.
32. A. Caracas, J. Altmann, “A Pricing Information Service for Grid Computing,” MGC2007, 5th International Workshop on Middleware for Grid Computing, Newport Beach, California, USA, November 2007.
33. VMWare Server, http://www.vmware.com/products/server/, 2009.
34. Weng, C.,Lu, X., Xue, G., Deng, Q., Li, M. A double auction mechanism for resource allocation on grid computing systems, In GCC, page 269, 2004.
35. Kant, U. and Grosu, D. 2005. Double Auction Protocols for Resource Allocation in Grids. In Proceedings of the international Conference on information Technology: Coding and Computing (Itcc'05) - Volume I - Volume 01 (April 04 - 06, 2005). ITCC. IEEE Computer Society, Washington, DC, 366-371.
36. Weng, C., Li, M., and Lu, X. 2007. Grid resource management based on economic mechanisms. J. Supercomput. 42, 2 (Nov. 2007), 181-199.
37. European Energy Exchange, http://www.eex.com/en/, 2009.
38. Enomaly, http://www.enomaly.com/, 2008.
39. Zimory, http://www.zimory.com/, 2009.
40. Zimory restrictions, http://www.zimory.com/index.php?id=33#c94, 2009.
41. A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. Web Services Agreement Specifcation (WS-Agreement), Version 2005/09, http://www.ggf.org.
42. Amazon, EC2 SLA, http://aws.amazon.com/ec2-sla/, 2009.

43. iX magazine. Sun Grid weltweit verfügbar. http://www.heise.de/ix/Sun-Grid-weltweit-verfuegbar-Update--/news/meldung/89194, May 2007.
44. Amazon EC2 usage restrictions, http://aws.amazon.com/agreement/#4b.
45. fluid Operations, http://www.fluidops.com, 2009.
46. H.K. Bhargava and A. Bagh. Tari_ Structures for Pricing Grid Computing Resources. In Gecon 2006, 2006.
47. EGEE DGAS: Price Authority. August 2007. https://edms.cern.ch/file/571271/1/EGEE-DGAS-PA-Guide.pdf.Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195--197 (1981)

TEMEP Discussion Papers

2009-01: Tai-Yoo Kim, Almas Heshmati and Jihyun Park, "Perspectives on the decelerating agricultural society".

2009-02: Yunhee Kim, Jeong-Dong Lee and Almas Heshmati, "Analysis of Pay Inequality and its Impacts on Growth and Performance in the Korean Manufacturing Industry".

2009-03: Chul-Yong Lee and Jongsu Lee. "Demand Forecasting in the Early Stage of the Technology's Life Cycle Using Bayesian Update".

2009-04: Jongsu Lee and Chul-Yong Lee, "A Forecasting Model Incorporating Replacement Purchases: Mobile Handsets in South Korea's Market".

2009-05: Ki H. Kang and Jina Kang, "Revisiting Knowledge Transfer: Effects of Knowledge Characteristics on Organizational Effort for Knowledge Transfer".

2009-06: Ki H. Kang and Jina Kang, "Does partner type matter in R&D collaboration for product innovation?".

2009-07: Ki H. Kang and Jina Kang, "How Do Firms Source External Knowledge for Innovation?: Analyzing Effects of Different Knowledge Sourcing Methods".

2009-08: Ki H. Kang and Jina Kang, "Do external knowledge sourcing methods matter in service innovation?: analysis of South Korean service firms".

2009-09: Gunno Park and Jina Kang, "The effects of teacher firms' characteristics and student firms' absorptive capacity on firm performance in technology alliances".

2009-10: Donghyun Oh and Almas Heshmati, "A Sequential Malmquist-Luenberger Productivity Index".

2009-11: Tausch Armo and Almas Heshmati, "Re-Orient? MNC Penetration and Contemporary Shifts in the Global Political Economy"

2009-12: Donghyun Oh, Almas Heshmati and Hans Lööf, "Technical Change and Total Factor Productivity Growth for Swedish Manufacturing and Service Industries"

2009-13: Tai-Yoo Kim, Almas Heshmati and Jihyun Park, "The Faster Accelerating Knowledge-based Society"

2009-14: Junseok Hwang, Jörn Altmann and Kibae Kim, "The Structural Evolution of the Web2.0 Service Network"

2009-15: Alireza Abbasi, Jörn Altmann and Junseok Hwang, "Evaluating Scholars Based on Their Academic Collaboration Activities: The RC-Index and CC-Index for Quantifying Collaboration Activities of Researchers and Scientific Communities"

2009-16: Dong-hyun Oh, Hans Lööf and Almas Heshmati, "The Icelandic Economy: A victim of the financial crisis or simply inefficient?"

2009-17: Sungki Lee, Donghyuk Choi and Yeonbae Kim, "Contextual effects on the

complementarities between R&D activities: An empirical analysis of the Korean manufacturing industry"

2009-18: Younghoon Kim, Yeonbae Kim, and Jeong-Dong Lee, "Corporate Venture Capital and Its Contribution to Intermediate-Goods Firms in South Korea"

2009-19: Yunhee Kim, Jae Young Choi, and Yeonbae Kim, "Complementarity and Contextuality in the Adoption of Information Systems in Korean Firms"

2009-20: Marcel Risch and Jörn Altmann, "Enabling Open Cloud Markets Through WS-Agreement Extensions"