On the Use of Resource-level Information for Enhancing SLA Negotiation in Market-based Utility Computing Environments

Mario Macías
Universitat Politecnica de Catalunya
Master in Computer Architecture, Networks and Systems
Master Thesis
Advisor: Jordi Guitart

June, 2009

Departament d'Arquitectura de Computadors
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Acknowledgements

I must be grateful to my advisor, Jordi Guitart, for its patience and unvaluable guide during the creation of this Masters Thesis.

This work is supported by the Ministry of Science and Technology of Spain and the European Union (FEDER funds) under contract TIN2007-60625 and Commission of the European Communities under IST contract 034286 (SORMA).

Abstract

Market-based Utility Computing arises as an efficient way of organisation of resources in Utility Computing scenarios. In Utility Computing markets, brokers that represent both clients and service providers meet in a market and negotiate for the sales of resources or services. This thesis defends the idea that efficient negotiations require of the usage of resource-level information for increasing the accuracy of negotiated Service Level Agreements and facilitating the achievement of both performance and economic goals. A negotiation model based on the maximisation of non-additive utility functions that considers multiple goals is defined, and its validity is demonstrated through the experiments performed.

Contents

Chapter 1

Introduction

1.1 Motivation

In the recent past years, academic and scientific entities as well as some companies owned big mainframes, clusters, or even supercomputers that had to be shared across their users for satisfying their computing requirements. Since they were expensive machines, the resources had to be managed having basically into account performance metrics: throughput, response time, load-balancing, etc. These systems were relatively easy to manage in a central way, since they had a single access point, and their usage was restricted to users of the same organization (or a few ones).

The big mainframes paradigm [1], where users own their computing resources, is being progressively transiting to a more utility-driven paradigm [2], where users do not own their resources and pay for the usage of remote resources. Utility computing has the advantages of other public utilities, like water or electricity. This is, the clients do not require spending neither an initial expenditure for the hardware nor maintenance costs (hardware, employees, electricity, physic space, cooling...).

The Grid [3] and, more recently, Cloud Computing [4] are the most promising current implementations of Utility Computing, the first in scientific and academic environments, and the second in the world of the enterprise. This new evolution has made the classical Resource Management mechanisms very inefficient because some reasons:

- The complexity of finding an optimum resource allocation is exponential, in huge systems like big clusters, data centres, or Grids, is growing very quickly [5]. Here are enumerated some causes:
  - A set of computers must be integrated via software and networking technologies.
  - Propelled by the transition from single- to multi-core processors, the average size of the clusters is being multiplied in few years. This implies more capacity and, as a consequence, more applications to manage.
  - New processing units are being added: GPGPUs, FPGAs, etc...
  - There is a transition from homogeneous resources and applications (mainly CPU-intensive) to heterogeneous resources and applications (web services, I/O intensive).

  - With the explosion of the implantation of Virtualization technologies [6], a single physical computer now becomes multiple logical computers, physical CPUs can be split into multiple logical CPUs in runtime, Memory can be resized dynamically, etc...
  - Grid and Cloud systems are not static: they are dynamically growing and decreasing, since new resources are hot plugged or removed.

- Multiple, remote users can now access the systems, and every one has its own preferences, which can be in conflict with the preferences of the other users.

- The idea of business is being introduced: some providers will sell their resources to the clients, which are willing to pay for accessing them. This introduces new high-level metrics: Quality of Experience, Quality of Business [7]... It is very difficult to manage resources having into account these metrics because they can be different for every provider and client, and the central resource manager does not have to know what good Quality of Experience of a user is.

- If the central resource manager breaks, the whole system gets useless. This is a big waste of resources in large systems.

Having into account these arguments, large systems seem to be too complex to be managed centrally. This thesis defends the usage of decentralized resource management as a paradigm to deal with the complexity. Concretely, market-based resource management is proposed by the next reasons:

- In utility computing, the possibility of doing business will motivate service providers to offer their resources in the system and give a Quality of Service (QoS) according to their real capacity.

- We can let the users reserve a spatial and temporary portion of the system, and we know that market mechanisms will obligate them to adjust their allocation to their real requirements.

- It is relatively easy to implement in a decentralized architecture.

- The complexity is reduced, because participants enter in the market looking for the satisfaction of their own necessities, and they do not need to know the global status of the system to maximize their utility.

Decentralized market-based resource management is starting to be implemented in open scenarios, such as Grid or Cloud computing, but this paradigm could also be implemented in closed systems, like Server Farms of a single organization, even when providers do not have business objectives. This thesis is not about making money, is about the usage of economic models (with real money or not) for an efficient resource organisation through large and very complex systems.

Figure 1 shows the generic logical architecture of a market-based resource allocation system. In this architecture, brokers that represent service providers or clients participate in a market to sell or buy their services. When the clients find there their

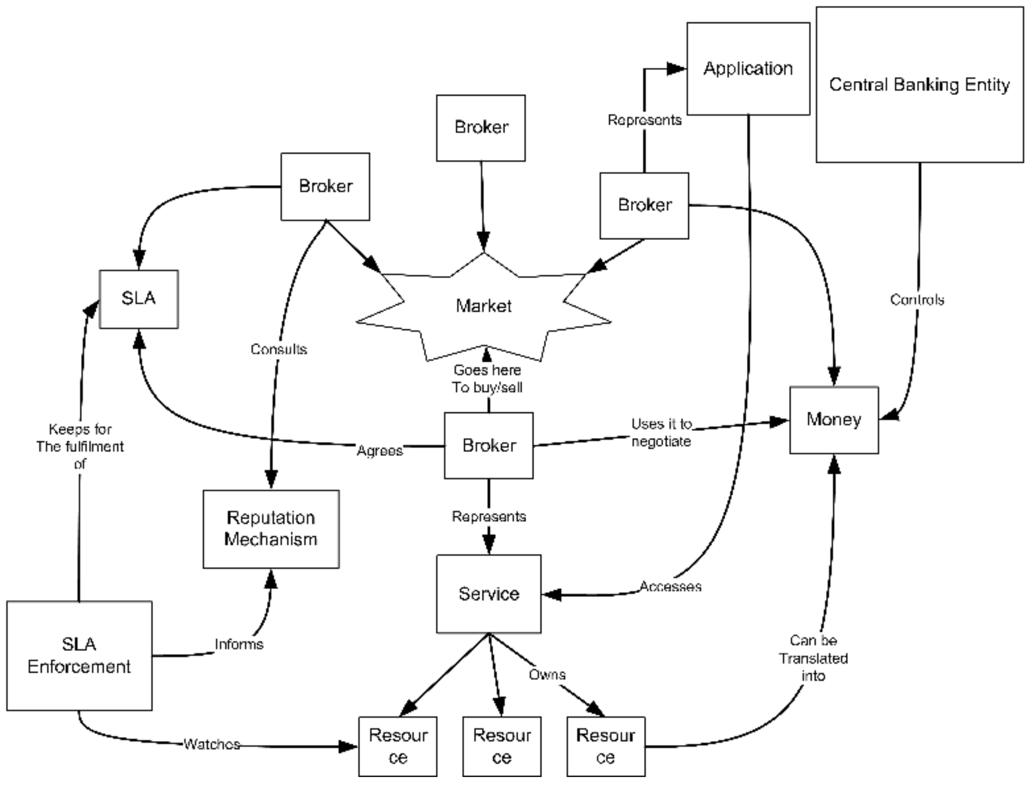

Figure 1.1: Market-based Resource Self-Organisation entities.

necessities, a **negotiation process** is started to establish the terms of the contract (QoS, price, time slot...). If both parts reach an agreement, the terms of the contract are specified in a Service Level Agreement (SLA) and the client application can acquire the bought resource. During the usage of the resources, the component called SLA Enforcement watches for the correct fulfilment of the terms of the SLA, and penalises the buyers or the sellers if they violate the SLAs.

1.2 Using resource information for improving negotiations efficiency

Since brokers that negotiate for the buying and selling of services are autonomous agents (this is, they communicate between them and take decisions without human intervention), it is needed to provide them some economic models and intelligent behaviour so they are able to take the best decisions for their represented actors in the market (client applications or service providers) and maximise their utility (see section 5.2.2 in chapter 2, dedicated to background concepts).

Examples of this economic behaviour are client classification, dynamic pricing, negotiation strategies, revenue maximisation, and so on. This thesis does not intend neither to evaluate them from an economic perspective nor invent new economic models and strategies; but some of these models will be explained and used for implementing negotiation scenarios and evaluate their performance.

This thesis uses existing economic models for negotiation and applies them to the negotiation of services and resources between computing agents: when the resource broker negotiates an SLA with the broker of a client, it has into account some economic terms, such as price, penalties for contract violation, time, etc. But there are other terms in the SLA, which are more technical, and also can have influence in the economic terms, specially those related with the Quality of Service (QoS) (e.g. throughput, response time...) or those related with the sales of plain resources (number of CPUs, speed, memory...).

For a purely-economic resource broker, it is very difficult to quantify the terms of the SLAs, since it has not enough technical knowledge about the status and punctual capacities of the resources. This can be seen also in real world sales, where brokers need to know not only the main characteristics of the products but also, for example, the capacity of production for a specified good in a concrete time, or the associated costs to the production and distribution process for a product that is customized for a concrete client. To illustrate this, imagine a factory of cookies: there are several sellers that look for potential clients, offering their products to be sold, for example in small shops, and also offering their production resources for creating cookies for other companies, for example those generic brands for big supermarket chains. When the sellers of the factory negotiate the terms of the contracts, they should talk first with the people that controls and schedule the production, for example:

- When the amount of production and the deadlines are negotiated, brokers must have knowledge about the production capacity of the factory, and about the reserved production for other current clients, in order to not order too many cookies than they actually are able to produce.

- When the price is negotiated, brokers have more liberty of election, but they also need to know the costs of production in function of the ingredients, the energy, the number of workers involved in the production, their salary, etc. And these costs can vary in time (due to the fluctuations in food and energy prices).

This example illustrates how a sales broker need to communicate with the agents involved in the production of goods in order to negotiate accurate contracts and get profitable sales. This thesis defends the idea that Market-Based Utility Computing is not an exception.

1.3 Contributions

Having in mind the content described in this chapter, this thesis deals with two questions:

- How the economic models can help resource management to be more efficient.

- How the low-level resource information can improve the accuracy of the economic models.

Summarizing, the main contributions of this thesis are:

1. Modelling and characterisation of the negotiations required to perform sophisticated sales in Market-Based Utility Computing in function of the desirable objectives, that also will be defined and studied.

2. Evaluation of the proposed economic models for the negotiation between brokers for the sale of computing utilities. This includes the comparison of several values for the parameters of the model and the evaluation about its feasibility and influence in the achievement of desired objectives.

3. Usage of low-level dynamic knowledge, provided by the resource fabrics, for giving support to economic negotiations. The required knowledge is defined by the contributions enumerated in point 1 and 2, and is acquired in real-time from the monitorisation of the Resource Fabrics.

1.4 Structure of the work

As introductory part, chapter 2 introduce some theoretical concepts that will help to understand better the work performed in this thesis. Chapter 3 cites other articles, thesis or projects related with economics applied to Utility Computing, in which this thesis is based.

The main contribution of this thesis starts in chapter 4, where the scenario and its particularities are defined. Chapter 5 shows the economic models, the components created, and describes the simulation where these models and components are evaluated. Chapter 6 shows and comments the results of the simulations. And, finally, chapter 7 comments the conclusions of the work, and points at future research lines.

Chapter 2

Background knowledge

This section introduces alphabetically some background concepts that reader should slightly know in order to understand correctly the theory covered in this thesis, and take the most of the explanation of the concepts and results.

2.1 Dynamic pricing

Economic markets are composed of a set of tradable goods (supply) offered by the market, and a set of buyers (demand) who ask for a subset of the supplied goods. The goal of dynamic pricing is to find an efficient allocation of the supply among the demand.

Law of Demand states that the higher price of a good, the less people will demand this good, and *Law of Supply* states that the higher price has a good, the higher quantity will be supplied, because selling higher amounts of goods at higher prices will increase the revenue. Having this into account, the supply and demand can be controlled by changing the prices of the goods: at lower prices, supply is low and demand is high, and at higher prices, supply is high and demand is low. This leads the market to a disequilibrium point.

In a market with disequilibrium, if the prices are too low the demand can not be satisfied because the supply is too scarce, and if the prices are too high the supply can not be sold completely because demand is too low. The efficient allocation that dynamic pricing pursues is achieved when it reaches the **equilibrium** point, and both demand and supply are satisfied.

Once shown the importance of pricing for finding the equilibrium of a market, two approaches to find the optimal pricing will be explained: *tatonnement* and *auctioning* [8].

2.1.1 Tatonnement

Tatonnement comes from a French word which means "trial and error", is an iterative method to progressively adapt the prices to the current market status. Ferguson et al. [8] propose an algorithm to do this process:

1. Choose an initial price vector $\vec{p} = \{p_1, p_2, \ldots, p_n\}$, where $p_1, p_2, \ldots, p_n$ are the prices for the resources numbered between 1 and n respectively, and a

minimum price vector $\vec{m} = \{m_1, m_2, \ldots, m_n\}$, which is the price that under this level the provider won't sell the resource.

2. For each resource, find the *Excess Demand Function* $Z_i(p)$, which is the demand for a resource i at a given price p_i minus the supply of it.

3. If for each resource i, $Z_i(p) = 0$ or $Z_i(p) \leq 0$ and $p_i = m_i$, equilibrium has been reached and the iteration stops.

4. Otherwise, for all the resources update the price vector $\vec{p}$ following the next formula: $p_i = Max\left[p_i + p_i \frac{Z_i(p)}{S_i}\right]$, where Si is the supply for the resource i.

5. Go to step 2

2.1.2 Auctions

In auctions, the price can be initially established by the seller, but the final price is determined by the customer who wins the auction in base to specified rules. These rules can vary in function to the kind of auction performed:

English auction The seller gives a start price, and buyers who are interested on acquire the resource increments the price in his bid. The highest bidder obtains the resource.

Dutch auction The seller gives the highest price, and it is gradually lowered by the seller until one of the buyers claims the resource.

Hybrid auction Asking price of a resource is increased if a bid is submitted, and decreased if no bid is submitted.

Sealed bid auction each customer submits a sealed bid, without knowing the bids from other customers and vice-versa. When all the bids are received, the seller gives access to the resource to the highest bidder. From the point of view of customers, when they arrive to the market, performs these steps to purchase the resources:

1. Compute their budget set
2. Find their most preferred elements (demand set)
3. Generate bids for the demand set

Double action Potential buyers submit their bids and potential sellers simultaneously submit their ask prices to the auctioneer, and then the auctioneer chooses some price p that clears the market: all the sellers who asked less than p sell and all buyers who bid more than p buy at this price p.

2.2 Futures market

A futures market is a central financial exchange where people can trade standardized futures contracts; that is, a contract to buy specific quantities of a commodity or financial instrument at a specified price with delivery set at a specified time in the future.

Futures traders are traditionally placed in one of two groups: *hedgers*, who have an interest in the underlying commodity and are seeking to hedge out the risk of price changes; and *speculators*, who seek to make a profit by predicting market moves and buying a commodity "on paper" for which they have no practical use. Hedgers typically include producers and consumers of a commodity.

For example, in traditional commodity markets, farmers often sell futures contracts for the crops and livestock they produce to guarantee a certain price, making it easier for them to plan. Similarly, livestock producers often purchase futures to cover their feed costs, so that they can plan on a fixed cost for feed. In modern (financial) markets, producers of interest rate swaps or equity derivative products will use financial futures or equity index futures to reduce or remove the risk on the swap.

The social utility of futures markets is considered to be mainly in the transfer of risk, and increase liquidity between traders with different risk and time preferences, from a hedger to a speculator for example.

2.3 Game Theory

Game Theory [9] is an applied branch of mathematics which tries to describe and evaluate the strategic situations where the individual success of the choice of an entity depends on the choices of the other entities (called players), or the status of its environment. The player is not able to know with complete certainty neither the decisions of the other entities or the environment.

For example, imagine that a smuggler proposes you the next business: you two will meet in a hidden forest. He will exchange its merchandise by your money. Since it is an illegal trade, it must be a very quick operation, so you will give him a bag with the money and pick up his bag containing the goods without spending time on checking the contents of the bags. In this game, you and the smuggler have two choices: to be honest or to cheat the other by giving bags filled by something without value. Game Theory tries to solve which is the best choice for you, having into account the possible choices of the smuggler.

Games can be classified in base to several criteria. Here are enumerated some of the most relevant:

Cooperative or non-cooperative A game is cooperative if the players are able to form binding commitments. For instance, the legal system requires them to adhere to their promises.

Zero-sum and non-zero-sum In zero-sum games, the choices of the players can neither increase nor decrease the available resources. That means that when a player gets benefit for its choice, the other will have a symmetric damage.

Repetitive and non-repetitive A game is repetitive when players have to play the same game several times. This will have influence in the taken decissions, because reputation is important in repetitive games, but not in non-repetitive games (a player can cheat and do not suffer consequences in the future).

2.4 Spot market

A spot market is a commodities or securities market in which goods are sold for cash and delivered immediately. Contracts bought and sold on these markets are immediately effective. The spot market is also called the "cash market" or "physical market", because prices are settled in cash on the spot at current market prices, as opposed to forward prices. An example of spot market is a supermarket, where goods are acquired and paid immediately.

2.5 Utility functions

This concept differs from the utility concept used in *utility computing* that is explained in section 1. In an economic system, the **utility** is a measure of the relative satisfaction that a participant obtains for its participation in the system (for example, the satisfaction of buying and consuming a determined good for a given price).

For quantifying and modelling the utility of an economic actor, **utility functions** are defined, which are helpful in the take of decisions, since they allow to know if a hypothetic situation is beneficial or not. For example, the utility function of a people that can buy the maximum of apples at the minimum price could be: $U = \frac{\#apples}{price}$. In a framework where participants act selfishly (like some market models), the objective of every actor is to maximize their own utility.

2.6 WS-Agreement

WS-Agreement [10] is a Web Services protocol for establishing agreement between two parties, such as between a service provider and consumer, using an extensible XML language for specifying the nature of the agreement, and agreement templates to facilitate discovery of compatible agreement parties. The specification consists of three parts which may be used in a composable manner: a schema for specifying an agreement, a schema for specifying an agreement template, and a set of port types and operations for managing agreement life-cycle, including creation, expiration, and monitoring of agreement states.

Figure 2.1 shows the conceptual parts of a WS-Agreement document:

Name The optional identifier for the agreement

Context Includes some metadata about the agreement, such as the participants, the services to agree, the duration, etc.

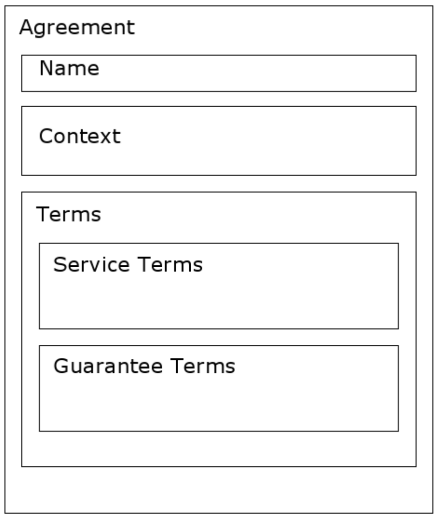

Figure 2.1: Structure of an Agreement

Service Terms There are one or more service terms that provide information needed to instantiate or identify a service to which the agreement belongs to, and to which guarantee terms can apply.

Guarantee Terms There are zero or more guarantee terms that specify the service levels that the parties are agreeing to. These terms are used to monitor the service and enforce the agreement.

In order to start a negotiation, a client or a provider makes an offer (in the format of an agreement as explained before) and sends it to an agreement Factory. The Factory advertises to the initiator the types of offers it is willing to accept by means of an agreement *template*. This template has the same contents that an agreement, plus some *Agreement Creation Constraints*, which are some optional elements that provide constraints on the values that the various terms may take in a concrete agreement.

Once the initiator receives the agreement template, it creates an agreement proposal that is sent to the other part. Then, the receiver of the agreement decides if he accepts or rejects it. The acceptance model ends here; this is, if the agreement is rejected, the negotiation ends and the initiator must create a new agreement proposal and send it once more.

Chapter 3

Related work

3.1 Negotiation models and protocols

Howard Raiffa established and compiled the mathematical basis of the negotiation models in his book *The Art and Science of Negotiation* [11]. This book classifies the different negotiation models in base to the characteristics of the environment and the negotiated goods. It will be widely referenced in this thesis, and in the most of the other works about negotiation.

Faratin et al. [12] applied and extended some existing models for service-oriented decision functions in bilateral negotiations between autonomous agents. It concentrates in many-parties, many-issues, single-encounter negotiations with an environment of limited resources, which is a variation of the *bilateral negotiation model* introduced by Raiffa: let $i \in \{a, b\}$ the negotiating agents and $j \in \{1, \ldots, n\}$ the issues under negotiation, let $x_j \in [min_j, mmax_j]$ be a value for issue j in the range of its acceptable values. Let ω_j^i the importance of issue j for agent i, and $\forall i, \sum_{1 \leq j \leq n} \omega_j^i = 1$. If $V_j^i : [min_j, max_j] \to [0, 1]$ gives the score that agent i assigns to a value of issue j in the range of its acceptable values, it is possible to define a scoring function of an agent for a contract, that is, for a value $x = (x_1, ..., x_n)$:

$$V^i(x) = \sum_{1 \leq j \leq n} \omega_j^i V_j^i(x_j) \tag{3.1}$$

Since computing services are qualitative in nature rather than quantitative, Faratin extends this model by adding qualitative values and associates fuzzy sets to them [13] in order to express better the quality in the negotiations.

Once the agents have determined the set of variables over which they will negotiate, the negotiation process between two agents consists of an alternate succession of offers and counter offers of values for the x, until an offer or counter offer is accepted by the other side or one of the parties terminates the negotiation. Faratin et al. demonstrated what this thesis affirms in chapter 1: negotiation tactics must be responsive to changes in the environment. Several tactics must be applied and dynamically changed in a same negotiation. They define *time-dependent tactics* (the acceptance value depends on the remaining negotiation time), *resource-dependent tactics* (the scarcer is the resource, the more urgent is the need for an agreement), *resource-estimation tactics* (the agent becomes progressively more conciliatory as the quantity of resource diminishes) and *behaviour-dependent tactics* (the next offer is computed based on the previous attitude of the negotiation opponent, useful for co-operative problem-solving negotiation settings).

By some experimental simulations, they proved that agents negotiating use their model were guaranteed to converge on a solution in a number of well defined situations and, with respect to tactics, they also discovered that: (i) irrespective of short of long term deadlines it is best to be a linear type tactic, otherwise and imitative tactic; (ii) tactics must be responsive to changes in their environment; and (iii) there is a trade-off between the number of deals made and the utility gained which is regulated by the initial offers.

The work in this thesis tries to extend the model of Faratin by extending the information extracted from the resources and used in the negotiation, and by having into account other economic factors, such as reputation, risk management, etc. The other main difference is that the work of Faratin was limited to a concrete scenario: client and provider brokers meet to negotiate for a concrete type of resource. The work in this thesis must consider the service discovery (this is, a market place) and the fact that agents can negotiate for a huge range of services.

Ouelhadj et al. [14] introduce a protocol for robust scheduling in Grid Computing based in the Contract Net Protocol [15]. The described architecture is similar to the current Grid Market systems, but it has the particularity that the SLAs are negotiated at two levels:

Meta-SLA negotiation The User Agent (UA) requests the execution of jobs on the Grid and negotiates a Meta-SLA with the Super Scheduler (SS) agents. There is one SS agent per each entity that owns a resource pool. The Meta-SLA contains high-level description of jobs supplied by the user and may be refined or additional information may be added in the final agreed SLA at the end of the negotiation. Uncertainty in user requirements such as time and cost constraint is represented by Fuzzy numbers.

Sub-SLA negotiation Each resource in a resource pool has a Local Scheduler (LS) agent, which is the responsible for scheduling the jobs that arrive. In the Sub-SLA negotiation level, the SS agents negotiate sub-SLA with the LS agents. The SS agents decompose the meta-SLA into its low level resource attributes, sub-SLAs which contain low level raw resource description such as processes, memory, processors, etc.

Other interesting feature in the work of Ouelhadj et al. is the possibility of a re-negotiation of the SLAs. Re-negotiation is useful when considering some uncertainties: presence of high-priority jobs, changes in the QoS requirements, resource failures, etc.

This thesis took some ideas from the Meta-SLAs of Ouelhadj's work and from WS-Agreement (see section 2.6): SLA negotiations between client and provider brokers are performed by using high-level QoS metrics. In a lower level, an Economically Enhanced Resource Manager (see section 3.3) helps in the negotiation of the SLAs by decomposing the high-level SLOs into low-level metrics, to calculate if a particular service can fit in the resources, given their status.

Vulkan et al. [16] evaluate the efficiency of English Auctions in the negotiation for services in multi-agent environments. Like in this thesis, they assume that the negotiations are initiated by the client. In addition, they introduce a *pre-auction* protocol for allowing the provider to initiate an auction when the client doesn't do. The winner of this auction is offered to the client as "take it or leave it". The difference with this thesis is that they use an English Auction instead of a direct negotiation and the presence of pre-auction protocols. However, the way they represent the negotiation terms is very similar to this thesis: a price and a set of SLOs.

3.2 Research projects on Market-Based Utility Computing

This section describes some research project related with the market allocation paradigm applied to Utility Computing.

3.2.1 GridEcon

GridEcon [17] is an European Community-funded project that offers market place technology to allow many small providers to offers their resources for sale. It designs the technology that is needed to create an efficient market place for trading commoditized computing resources as standardized Virtual Machines (VM). The market mechanism used has been designed to be simple for participants and also economically sound. The later is concerned with inducing the right economic incentives to participants and avoiding unwanted strategic behaviour leading to market dominance with large players. The GridEcon project also designs a series of value-added services on top of the market place (e.g. insurance against resource failures, capacity planning, resource quality assurance, etc...), ensuring quality of the traded goods for Grid users.

When a buyer wants to acquire a resource, it sends a bid to the market by specifying its requirements according to the next terms:

- The type of resource (VM) required.
- The quantity of resources required.
- The start time of the interval for using the resources.
- The time duration of using the resources.
- The price expressed in €/min/unit.
- The time limit until which the bid is valid. If this time limit is reached without the bid being matched, the bid is removed from the system.

In order to keep the definition of the bid as general and flexible possible, instead of allowing only fixed values for the number of VMs and the time duration, it is allowed to specify whether these constraints should be met with equality, or $\leq$ or $\geq$. GridEcon allows both bids for futures and spot markets (see sections 2.2 and 2.4 respectively).

Resource providers can also send its spot and futures offers to the market, in order to describe what resources they are selling:

- The type of resource (VM) offered.
- The quantity of resources offered.
- The start and end time of the interval when the resources are available
- The price expressed in €/min/unit.
- The time limit until which the offer is valid. If this time limit is reached without the offer being matched, the offer is removed from the system.

Once bids and offers are in the market, a matching algorithm is used in order to help buyers to find the most suitable provider for them, and *vice-versa*. The trading is performed by means of a continuous double action (see section 2.1.2), where both bids and offers that are submitted by traders are placed in queues and ordered in decreasing order of price for bids, and in increasing order of price for offers. This will guarantee that the buyers that are available to pay more and the sellers that put their resources at lower prices will be the first on finding their matches.

3.2.2 SORMA

The Self-organising ICT Resource Management (SORMA) [18] is an EU IST [19] funded project aimed at developing methods and tools for efficient market-based allocation of resources, using a self-organising resource management system and market-driven models, supported by extensions to existing grid infrastructure. Topics addressed include Open Grid Markets, economically-driven middleware, and intelligent support tools.

Unlike traditional grid environments, jobs submitted to SORMA are matched with available resources according to the economic preferences of both resource providers and consumers, and the current market conditions. This means that the classic grid job scheduler, which is based on performance rules, is replaced by a set of self-organising, market-aware agents that negotiate Service Level Agreements (SLAs), to determine the 'best' resource allocation to fulfil both performance and business goals. In SORMA, an Economically Enhanced Resource Manager (EERM) [20] exists at each resource provider's site, and acts as a centralised resource allocator to support business goals and resource requirements.

This thesis is based in the work performed within the SORMA European project, concretely in the EERM component (see section 3.3). This component will combine the purely economic knowledge (because is in direct contact with the economic layers of SORMA marketplace) and the plain resources data (because it manages directly the resource fabrics) to help economic brokers to perform better negotiations and enforce the resource management, not only having into account performance but also economic goals.

This section describes the objectives and requirements for the enhancements, and then follows a description of the key mechanisms that are to be integrated in the Economically Enhanced Resource Manager (EERM) [21].

The main functions of EERM are:

- Decide whether incoming tasks are accepted or not, based of the usage of the resources for a given time slot, the client priority, and the sale price.
- Calculate prices for offered jobs and services based in current market status, current resource usage and predictions about the impact of a job in the usage of resources.
- Check that the accepted SLAs can be kept. If one or more SLAs can not be fulfilled it takes the decision to suspend or cancel jobs to ensure the fulfilment of the other SLAs and maximise overall revenue.
- It is responsible of the communication with the local resource managers and influences the local resource management to achieve a more efficient global resource use.

The behaviours in the previously enumerated functions can be tuned with policies.

Objectives and Requirements

The main goals of these enhancements are to link technical and economic aspects of resource management and strengthen the economic feasibility of the Grid. This can be achieved by establishing more precise price calculations for resources, taking usage of the grid, performance estimations and business policies into account.

The introduced mechanisms should be able to deal efficiently with the motivational scenarios given earlier. This means they have to feature *client classification*, different types of *priorities* for jobs from certain clients, *reservation* of a certain amount *of resources* for important clients, and *dynamic calculation of prices* based on various factors.

In addition to these requirements the system should also offer quality of service and be able to deal with situations in which parts of the resources fail. To adapt to different scenarios and business policies of different situations it should be highly flexible and configurable via policies.

When designing mechanisms various economic design criteria [22], [23] should be considered. These following criteria apply to the respective features as well as the overall system and the market mechanisms it is embedded in.

Individual Rationality. An important requirement for a system is that it is individual rational on both sides, i.e. both providers and clients have to have a benefit from using the system.

Simplicity and Computational Costs. While the enhancements introduce some additional factors and they should not introduce any unnecessary complexity. Similarly client classification, quality of service and dynamic pricing add some additional computational complexity, however they should not add any intractable problems and its benefits should outweigh its costs.

Revenue Maximization. A key characteristic for Grid providers is revenue maximization or more general utility maximization.

Incentive Compatibility. Strategic behaviour of clients and providers can be prevented if a mechanism is incentive compatible. Incentive compatibility means that no other strategy results in a higher utility than reporting the true valuation.

Efficiency. There are different types of efficiency. The first one considered here is pareto optimimality. An allocation is considered pareto optimal if no participant can improve its utility without reducing the utility of another participant. The second efficiency criterion is allocative efficiency. A mechanism is called allocative efficient if it maximizes the sum of individual utilities.

Key Features

The motivational scenarios and the further requirements lead to four key features of the EERM presented in this work.

Quality of Service The first feature is quality of service. This can be broken down into two aspects. The first aspect is to assure adequate performance during normal operation of the resources. Overload situations can lead to reduced overall performance [24] and thereby can result in breaking QoS agreements between the provider and clients. Thus it is necessary to have a mechanism that ensures that jobs will not be accepted if they result in an overload situation.

The second aspect of quality of service regards situations in which parts of the resources fail. To be able to fulfil all SLAs even in situations of partial resource failure it would be necessary to keep an adequate buffer of free resources. Where this is not feasible there should at least be a mechanism that ensures that those SLAs that can be kept with the available resources are fulfilled. This can be done by suspending or cancelling those jobs that can not be finished in time due to the reduced availability of resources.

Job Cancellation Related to QoS is the feature automatic job suspension and cancellation. It is needed to ensure quality of service in situations where problems arise, i.e. parts of the Grid fail or the estimations of the utilization were to optimistic. Cancellation of lesser important jobs to free capacity for incoming jobs with higher importance, i.e. jobs from a client with a higher classification or a jobs that deliver significantly more revenue is also possible [20, 25].

Job Migration Besides the job cancellation for ensuring the QoS of the jobs that deliver more revenue, Macías et al [20] propose the migration of tasks between the nodes of the resource pool, in order to allow a better usage of the free resources, by avoiding the fragmentation of them. For example, instead of having 4 nodes with one free CPU each one, they propose to redistribute the jobs allocation to try to have 1 node with 4 free CPUs.

Dynamic Pricing Another enhancement is dynamic pricing based on various factors. [26] shows an approach for a pricing function depending on a base pricing rate and a utilization pricing rate.

However the price can depend not only on current utilization but also on other factors such as projected utilization, client classification, projected demand, and reputation [27].

Pricing should also be contingent on the demand on the Grid market. This feature can be either implemented in the EERM or in a dedicated component responsible for trading. This agent would request a price based on factors including utilization of the grid and client classification from the EERM and then calculate a new price taking the situation on the market into account.

Client Classification. Earlier giving different privileges to clients was mentioned, e.g. by discriminating on factors like price and quality of service. This part describes the factors that can be used to differentiate various client classes.

Price Discrimination. Price discrimination or customer-dependent pricing is one way to differentiate between different classes of clients. One idea to achieve this is introducing Grid miles [28] in analogy to frequent flyer miles. Clients could be offered a certain amount of free usage of the Grid or a 10% discount after spending a certain amount of money.

Reservation of Resources. For certain users it may be very important to always have access to the Grid. This class of users could be offered a reservation of a certain amount of resources. One option is to reserve a fixed share of resources for a certain class of users another possibility is to vary this share depending on the usage of the system.

Priority on Job Acceptance. Another option is to give a class of client priority on job acceptance. When the utilization of the system is very low jobs from all classes of clients are accepted but when the utilization of the Grid rises and there is competition between the clients for the resources, jobs from certain clients are preferred. There can be two types of priorities: strict priorities and soft priorities.

Strict priority means that if a job from a standard client and a client with priority compete for acceptance, the job from the client with priority always wins. Jobs from clients with priority are always preferred, thus there is no real competition between the different classes of clients.

Soft priority means jobs from clients with priority are generally preferred but standard clients have the chance to outbid clients with priority. Thus soft priority is essentially a discount on the reservation price or bid that may only apply in certain situation, i.e. when utilization exceeds a certain threshold.

Quality of Service. Another factor where differentiation for classes of clients is possible is quality of service. For some classes of clients quality of service is offered, for others not. Offering different levels of quality of service for different classes of clients is also possible. An example for this would be offering different risk levels [29].

3.2.3 Grid4All

The Grid4All [30] project promotes the concept of a democratic Grid, accessible to modest groups of end-users such as schools, families, non-governmental or- ganizations, or small businesses. It enables to put together people and computing resources to form Virtual Organizations (VO): a virtual collection of users or institutions that pool their resources into a single virtual administrative domain for a common purpose. Virtual Organizations can also trade resources among different VO on a decentralized market place.

In the area of data services, compared to previous Grids, the Grid4All architecture provides with a minimal administration enhanced support for content sharing and collaboration within groups. Semantic search and ontologies are used to locate and select among diverse resources and services.

The most relevant components of Grid4All for this thesis are:

Information Service Provides match-making between semantic service descriptions and client requirements.

Resource Management Enables members to contribute resources to the VO and to discover and allocate VO resources.

Resource Brokerage Arbitrates and allocates Grid resources to VOs. VOs lease resources on need by negotiating at the resource market place. The Grid4All model of VOs is similar to peer-to-peer networked applications: members of a VO use resources that belong to the VO. Non-members can incorporate resources to adapt to fluctuations in supply and demand (see section 2.1). Market-based brokering with pricing mechanisms provides fair arbitration to do that, gives incentives and is decentralized.

Auction Servers Through a common market interface, participants can send bids specifying its necessities. The auction determines allocations and the transaction prices. Behind the same interface, multiple mechanisms are allowed.

Market Factory Allows the designers and developers to register implementation for new types of auction mechanisms. Also allows to the traders to choose a specific auction format and select a concrete auction server.

Market Information Service In decentralized markets, it is important to obtaining synthetic and summarized information, such as average prices or demand. This information is crucial for allowing market brokers to take correct decisions.

3.2.4 Catnets

The Catnets project proposes a market-based approach based on the Catallaxy concept [31]: a market order without planned ends, characterized by the *spontaneous order* which emerges when individuals pursue their own ends within a framework set by law and tradition. The function of government is to maintain the rule of law which guarantees fair and equal procedures, but is neutral as to goals.

The advantage of Callaxy is that does not need to support for centralized brokers: it uses a “free market” self-organisation approach, which enables prices within the market to be adjusted based on particular demands being placed on particular scarce services. To implement this decentralized and highly chaotic market, Catnets adopts a P2P approach, which allows establishing a symmetric interaction between peers, and allocate dynamically the communication paths in function of the changes in the network topology.

Catnets has a clearly-defined layered architecture, in function of the level of abstraction of their functionalities:

Application layer Is given by the domain-specific end user applications like collaboration tools, problem solving environments, and may others.

Economics Algorithms layer Implements economic algorithms for resource allocation that includes a set of interacting agent services that play the roles of Sellers and Buyers. This layer is domain and platform independent.

Economics Framework layer Offers primitives for supporting the implementation of catallactic algorithms, such as finding peer agents to negotiate, starting negotiation, making a bid, etc.

Peer Agent layer Platform that hosts the Catallactic agents offering a generic P2P application model with abstractions for the discovery and communication mechanism, and a generic interface with the underlying platform.

Base Platform layer Supports applications and Catallactic middleware. The interaction with the middleware depends on the architecture of the base platform, such as Globus [32] or UNICORE [33].

3.2.5 Nimrod-G

Nimrod-G [34] is a tool for automated modelling and execution of parameter sweep applications over global computational Grids. It provides a simple declarative parametric modelling language for expressing parametric experiments. A domain expert can easily create a plan for a parametric experiment and use the Nimrod system to submit jobs for execution. It uses novel resource management and scheduling algorithms based on economic principles. Specifically, it supports user-defined deadline and budget constraints for schedule optimisations and manages supply and demand of resources in the Grid using a set of resource trading services.

Nimrod-G provides a persistent and programmable task-farming engine (TFE) that enables “plugging” of user-defined schedulers and customised applications or problem solving environments (e.g., ActiveSheets) in place of default components. The task-farming engine is a coordination point for processes performing resource trading, scheduling, data and executable staging, remote execution, and result collation. In the past, the major focus of our project was on creating tools that help domain experts to compose their legacy serial applications for parameter studies and run them on computational clusters and manually managed Grids. It is focused on the use of economic principles in resource management and scheduling on the Grid in order to provide some measurable quality of service to the end user.

The key components of Nimrod-G resource broker consist of:

- Nimrod-G Clients, which can be:
  - Tools for creating parameter sweep applications.
  - Steering and control monitors.
  - Customised end user applications.

- The Nimrod-G Resource Broker, that consists of:
  - A Task Farming Engine (TFE).
  - A Scheduler that performs resource discovery, trading, and scheduling.
  - A Dispatcher and Actuator.
  - Agents for managing the execution of jobs on resources

3.3 Economic Enhancements Applied to Resource Management

To improve performance in the commercialization of distributed computational resources and increase the benefit of grid providers we propose the introduction of various economic enhancements into resource management [35].

Yeo et al. [36] propose a model for market-based cluster computer with some elements common to EERM. The cluster nodes are connected to a central manager which incorporates other sub-components for performing Pricing, job scheduling, monitoring and dispatching, user admission control, and so on. The main difference with EERM is that EERM is conceived to be integrated into a higher level grid market, so it does not implement some market functions such as accounting, billing, or identity provisioning.

Freedman et al. [37] focus on Peer-to-Peer (P2P) content distribution by identifying explicitly highly demanded files and rewarding most those peers sharing highly demanded content. They use a market-based mechanism for allocating network resources across multiple files and play with the Law of Offer and Demand for incentivising providers to sell most scarce high-value resources (only files, but not CPUs or Memory). Their system is also designed so that the buy client chooses files consistent with its best interests, since it seeks to download at the current minimum price.

Auyoung et al. [38] compare the utility of using auction-based schedulers with classical batch schedulers. They explain that some users are unwilling to accept the uncertainty of auctions; hence, they develop a buy-it-now mechanism that allows risk-averse users to instantly acquire resources at premium prices. They also describe, by some utilities comparison, how intelligent monetary policy, in particular the judicious use of a savings tax, ameliorates the budget disparities induced by the 90/10 usage patterns common in these environments.

Chapter 4

Problem statement

4.1 Scenario Definition

The first question that arises about the scenario of this thesis is: what kind of goods must be sold in the utility computing market? There are two alternatives: the first one, which comes from the Grid Computing area, is to sale plain resources, such as CPUs, memory, storage, or network bandwidth. This is the simplest option for the provider, since resource assignation is direct (it only needs to assign what the client bought) and the SLA enforcement only consists on assuring that these resources are always available to the client. The client is in charge of sending the software to the provider, install and configure it, and make usage of the resources.

The second option is related with the *Software as a Service* (SaaS) view: the provider only sells some services already installed in its resources. This is very simplistic for the end user: only needs to know the interface to access the service, and make remote service calls to get the required results. For the client it is only important **what** it gets, without taking care about **how** to get it, where is it placed, or how many resources are used to perform the service. The difficulty here is for the service provider: from a high-level description of the service, how to translate the QoS terms into plain resource thresholds? The so called SLA decomposition [39] tries to deal with this problem. SLA decomposition will be taken into account in this thesis, but no research will be performed in this field.

This thesis will consider the scenario of services sales. To not keep apart the classical Grid scenarios, the sale of plain resources is considered as a kind of high-level resource service, and the SLAs will allow the inclusion of the amounts of resources as high-level SLOs.

Figure 4.1 shows the sequence diagram of the scenario where this thesis takes place. Before the negotiation starts, the **EERM** of the Service Provider must register its offered services in the **Service Discovery component**. For each service, it is provided some semantic information that allows to identify what service is and its functionalities, and an extra meta-SLA with some data about the SLA terms (also known as Service Level Objectives) that the service provider is willing to negotiate, such as response time, throughput, quality (for example in video compression), duration of the service, etc.

When a **Client Broker** wants to acquire a service, it queries the Service Discovery by providing some semantic information, and the Service Discovery returns

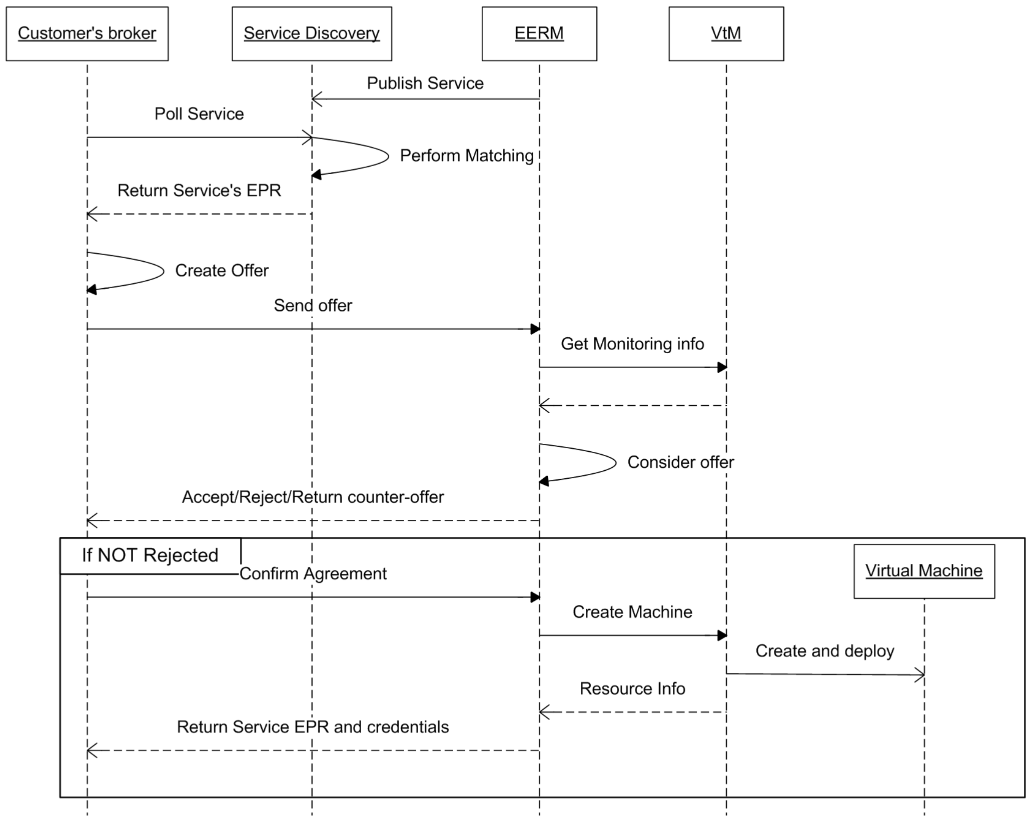

Figure 4.1: Workflow of the resource sale process

a list of the Service Providers that match the requirements (every provider has its own EERM) and the meta-data about the negotiable SLA terms.

The next process is similar to the one described in the WS-Agreement specification (see section 2.6), but with an extra iteration in the negotiation. Before starting the negotiation the Client Broker selects the most interesting services, and creates a proposal of agreement for each one; using the meta-data it creates an uncompleted SLA with its requirements, and leave other SLOs as void, to be completed by the EERM in the next negotiation process.

When the EERM receives the SLA proposal, it evaluates if the proposed terms can be accepted. There are two possible scenarios:

1. The SLA can be accepted: the EERM optionally fills the void SLOs in with some important data, such as price, time, etc, and sends the acceptance message to the Client Broker.

2. The SLA can not be accepted: the EERM sends a denial message to the Client Broker and the negotiation ends, or the EERM proposes a counter-offer, by modifying some of the proposed SLOs and filling in the other important SLOs that are incomplete.

If the Client broker received from the EERM an acceptance message or a counter-offer, it evaluates it and finishes the negotiation by not accepting the SLA or by sending a confirmation message to the EERM.

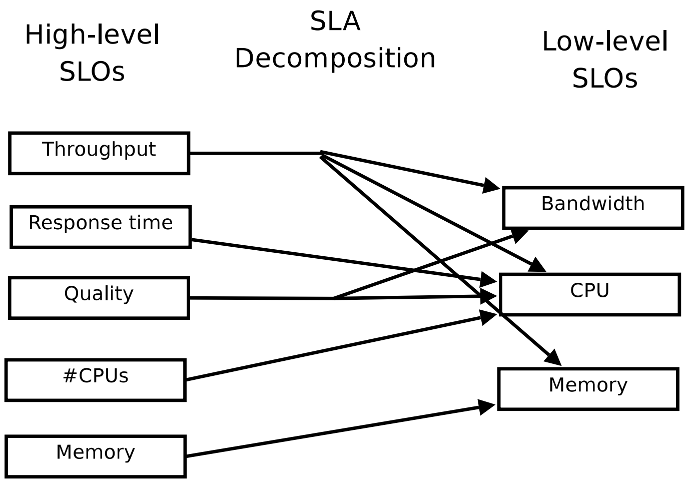

Figure 4.2: Example of SLA decomposition. High level SLOs are translated into low-level. In this thesis, when the plain resources are treated as high level-services, the same SLOs can be both high and low level

Once the agreement is confirmed, the EERM will ask the **Virtualization Manager (VtM)** [6] for the creation of a tailored **Virtual Machine** that has enough resources for fulfilling the agreed SLA. Note that the SLA decomposition process is performed at this point of the workflow: the EERM get high-level SLOs and must decompose them into plain resource metrics (see figure 4.2).

There are three classes of SLAs, listed by order of priority: gold, silver, and bronze. The SLAs with higher class will be more expensive than the other with lower class, but they are preferential over the others when a resource has failed and some tasks must be cancelled, and the provider will use more resources to minimize the probability of failure [25, 27, 20, 25].

4.1.1 Assumptions

This scenario is a very simplistic sales scenario, in contrast with current research about Utility Computing Markets. Since this thesis is focused in the negotiation process, the most of the basic components of a Market are ignored and assumed that they are present:

- There is already a service discovery mechanism where providers can offer their services, and clients can look for them by providing some semantic information. This component will be in the prototypes used for doing experiments in this thesis, but it will be very simplistic.

- The EERM has a SLA decomposition component that is able to translate from the Service Level Objectives (SLO) that compose the SLA, such as response time or throughput, to system-level thresholds, that are used to assign the correct number of resources to a service, for fulfilling the agreed SLA.

- All the communications are performed in a secure channel, and there is an Identity Provider that allows the components be sure that they are not negotiating with impostors, and to the service providers that the brokers that make use of its services are allowed to access them.

- There is a Payment component, where the client broker pays for the service after the negotiation is ended, and before the service is being used.

- There is a Market Information System that will allow the brokers of both clients and providers to know the status of the market; e.g, the prices that other customers are paying for similar services.

4.2 Characterisation of the negotiation

Once the scenario has been defined, it is time to define the content and details of the research of this thesis. Since it is focused in the usage of resource-level information for enhancing the SLA negotiation in utility computing markets, the first step to do is to characterize the negotiation. To do that, this thesis takes into account the *Organizing Questions* that Raiffa proposed in his book [11].

Are more than two parties? No. The only parties in the negotiation for a computing resource or services are both client and provider broker agents.

Are the parties monolithic? Yes. We assume that the brokers have been fully configured and can take the negotiation decisions by themselves, without need to negotiate first or later with its representeds.

Is the game repetitive? Yes. Once the negotiation is ended, a client can meet the provider for new negotiations. This means that reputation is present, and what one of the parties do in a negotiation can be taken into account in future negotiations.

Are there linkage effects? Short response: no. Explanation: *linkage* means that the conditions that one of the parties accept can be demanded in future negotiations by other brokers. For example, if a provider sells a resource by 10€and future clients know it, in the future they can demand the same resource at as much the same price. For having linkage efects, it would be needed a *Market Information System.* Since this information system is out of the scenario of this thesis, we will assume that there are **not** linkage effects.

Is there more than one issue? Yes. There is a large number of issues to negotiate, such as price, time and several SLOs.

Is an agreement required? No. That means that parties can break the negotiation since the agreement is not critical (this is, not reaching the agreement is not catastrophic).

Is ratification required? No. As it is explained before, brokers are fully capable to take autonomous decisions and do not need to ratify them with their represented entities.

Are threats possible? Yes. There are some fixed threats. For example, if there is no agreement, the client will look for another provider and the provider will not make money by the sale. Other type of threat is that if the provider oversells its resources, there is the threat of not being able to offer all of them, breaking the SLAs, loosing money due to the penalizations and being decreased its reputation in the market.

Are there time constraints or time-related costs? Depending on the particular situation of a concrete party, it can be. But these hypothetical possibilities are too extensive, complex and difficult to know by the opposite party and this thesis will assume that **no**.

Are the contracts binding? Yes. And there are entities such as *SLA Enforcement* to check at any moment that the contracts are being fulfilled and, if not, make the violating party to pay a penalisation.

Are the negotiations private or public? Private. No other brokers but involved ones will know about the current negotiation status.

What are the group norms? The involved parties are *cooperative antagonists.* Such disputants recognize that they have differences of interests; they would like to find a compromise, but they fully expect that the other party will be primarily worried about their own interests. They do not have malevolent intentions, but neither are they altruistically inclined. They are slightly distrustful of one another; each expects the others to try to make a good case for their own side and to indulge in strategic posturing. They are not confident that the others will be truthful, but they would like to be truthful themselves, within bound. They expect that power will be used gracefully, that all parties will abide by the law, and that all join agreements will be honoured.

Is third-party intervention possible? No. There is no intervention of any mediator nor arbitrator.

4.3 Research perspective

Once the negotiation scenario is defined and the negotiation itself is characterized, it is time to describe the research perspective of this thesis.

Since the motivation of this scenario is the enhancement of resource management in service providers, the research about negotiations in utility computing markets will be performed in a *descriptive* way from the point of view of market clients, and in a *prescriptive* manner for the providers. *Descriptive* means that the behaviour of clients will be analyzed and described in order to *prescribe* to the resource manager the best actions to do for maximising its own utility.

Chapter 5

Description of Work

5.1 Architecture of the EERM

This part describes the architecture of the EERM that was designed based on the requirements and key features of section 3.3. It includes a description of the components with the aid of sequence diagrams. An overview of the architecture of the EERM can be seen in Fig. 5.1.

The EERM interacts with various other components, namely a Grid Market Middleware, a Monitoring component and the Resource Fabrics. The Grid Market Middleware represents the middleware responsible for querying prices and offering the services on the Grid market. The Monitoring is responsible for monitoring the state and the performance of the Grid and notifying the System Performance Guard in case of problems. Additionally data collected by the Monitoring is used by the Estimator component to for its predictions. Resource Fabrics refers to Grid Middlewares such as Condor [1] or Globus [32].

5.1.1 Economy Agent.

The Economy Agent is responsible for deciding whether incoming jobs are accepted or not and for calculation prices for jobs. These can be used both for negotiation as well as bids or reservation prices in auctions.

First the Economy Agent receives a request from a market agent. Then it checks whether the job is technically and economically feasible and calculates a price for the job based on client category, resource status, economic policies and predictions of future job executions from the estimator component. If necessary the Economy Agent can also invoke the System Performance Guard to free capacity for more important jobs.

5.1.2 Estimator.

The Estimator component calculates the expected impact on the utilization of the Grid. This component is based on the ideas introduced in [40]. The Estimator component is also important to prevent reduced performance due to overload [24].

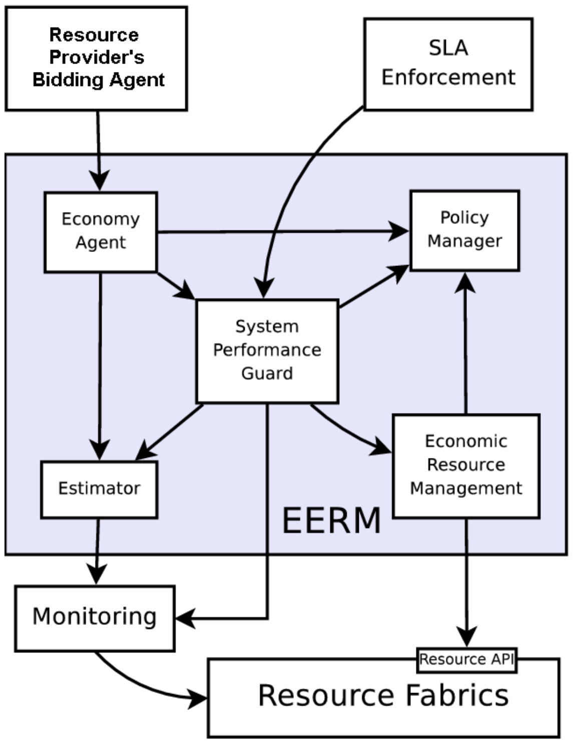

Figure 5.1: EERM Architecture

5.1.3 System Performance Guard.

The System Performance Guard is responsible for ensuring that the accepted SLAs can be kept. In case of performance problems with the resources it is notified by the Monitoring component. After checking the corresponding policies it determines if there is a danger that one or more SLAs cannot be fulfilled. Then takes the decision to suspend or cancel jobs to ensure the fulfilment of the other SLAs and maximize overall revenue. This is done in accordance to policies concerning client classification. Jobs can also be cancelled when capacity is required to fulfil commitments to preferred clients.

5.1.4 Policy Manager.

To keep the EERM adaptable the Policy manager stores and manages policies concerning client classification, job cancellation or suspension, etc. Policies are formulated in the Semantic Web Rule Language (SWRL) [41]. All features of the EERM require the respective components to be able to communicate with the Policy Manager and base their decisions on the corresponding policies.

A simple example from a pricing policy in SWRL is the following rule which expresses that if the utilization is between 71% and 100% there is a surcharge of 50:

$$Utilization\,(?utilization) \wedge InsideUtilizationRange\,(?utilization, "71\% - 100\%") \\ \Rightarrow SetSurcharge\,(?utilizationsurcharge, "50")$$

5.1.5 Economic Resource Management.

The Economic Resource Management is responsible for the communication with the local resource managers and influences the local resource management to achieve a more efficient global resource use.

5.2 On the usage of non-additive utility functions

The first issue that must be defined is the analytic model for representing the negotiations that will be performed by the *Economy Agent* with the help of *System Performance Guard*, *Estimator* and *Policy Manager* components. *Economy Agent* takes the economic decisions in function to the monitoring data that receives from *System Performance Guard*. *Estimator* component will calculate the impact of a future job for negotiating the required amount of resources to sell for the execution of a service. Instead of *Economy Agent* is the brains and the communication point for the negotiation operator, it acts in function to the policies or utility functions stored in the *Policy Manager*.

This model must take into account the negotiated SLOs and other terms expressed in section 3.2.2, such as client classification or reservation slots plus the sale price. The model proposed in this thesis is the *"two parties, many issues"* model proposed by Raiffa and Faratin (see equation 3.1).

This model is pretty easy to manage and calculate the maximum and minimum utilities: during the negotiation process, the provider can know easily which factors addends to increase or decrease in order to increase the utility and keep it the nearest possible to the maximum. However, it has a problem: it is an *additive model* which assumes that all the factors are independent from the others. This is not true, since some factors such as the price is strongly related with the SLOs, the type of client, etc.

Let S be the SLA under negotiation, the **non-additive utility** function U used in this thesis for the service provider is the next:

$$U(S) = \sum_{i=1}^{m} o_i u_i(S) \tag{5.1}$$

Where m is the number of goals for the provider, such as revenue maximisation, high reputation, performance maximisation, high occupation of resources, satisfaction of certain type of users, etc. u_i is the sub-utility function that defines how much will be the objective i satisfied, and o_i is a number between 0 and 1 that defines the priority that the provider assigns to the concrete objective. It must be considered that $\sum_{i=1}^{m} o_i = 1$.

Instead equation 5.1 is similar to an additive function, actually it is not. Instead of calculating each of the sub-utility functions in function of a single SLA term and finally add them up, equation 5.1 calculates all the sub-utilities in function of the whole SLA. This is because the different objectives are not independent from the others and, for example, revenue maximisation can affect negatively the client satisfaction. Following this example, an additive function will suppose that increasing the revenue will always increase the general utility, but that might require to break some agreed SLAs [20, 25] and in consequence, decrease the general user satisfaction. If the weight assigned to the user-satisfaction goal is higher than the assigned to revenue maximisation, the global utility will decrease instead increase.

The usage of non-additive utility functions will allow a more accurate definition of the utility; however, finding the maximum utility is much more difficult to do analytically. The literature for dealing with this problem is not so extensive like in the additive case. Stamoulis and colleagues [42] develop a negotiation framework for telecommunication services. However, this approach does not fulfil the requirements of this thesis, since the only and main objective of the provider is the economic revenue. Angilella et colleagues [43] use Choquet integrals [44] for helping a decision maker to choose the best option from a finite set of alternatives, based on a finite set of criteria. Neither this approach fulfils the requirements of this work, because the EERM must "create" a best alternative for a proposal, not to choose it from a set of proposals. The set of alternatives of the service provider is infinite.

Another option is to approximate the maximisation of such non-additive utility functions by using self-learning techniques such as Neural Networks [45]. Such networks should take an input SLA with some blank SLOs and create an output SLA with the next characteristics:

- The global Utility must be positive and sub-optimum.
- The SLA must be acceptable for the parts. This implies that the counter-offers that the EERM can perform must not be too different from the initial offers of the clients.
- The SLA must be realistic, this is, the provider must be able to fulfil it.
- It is desirable that, when a complete offer without blank SLOs is received, the EERM is also capable to create a new counter-offer, similar to the received one, but increasing its utility.

Modeling a self-learning system that fulfils all of these requirements is by itself a complete research line and will not be treated in this thesis. Next subsection will describe the approximation that is performed for the experiments in this work.

5.2.1 Negotiation model

One of the main contributions of this thesis is the usage of non-additive utility functions. However, there will be some limitations in order to make the modelling and maximisation of the functions more affordable:

- If the utility of a received offer is positive, the offer will be accepted.
- There is allowed only one blank SLO in the received offers.
- When an offer whose SLA can not be fulfiled is received, a counter-offer is generated. However, only SLOs related to time or price are changed (the other QoS terms are kept as the original ones).

The two last restrictions will allow the easy isolation of the SLA terms and creating counter-offers by only using inequations

5.2.2 Negotiation terms and Utility functions

First is needed to define the set O of objectives, the set S of SLA terms and the utility function $U(S)$ that calculates how beneficial the proposed SLA for the objectives of the provider is.

Before defining the set of objectives and utility functions, some representative objectives must be chosen for a service provider. Concretely for this thesis, four objectives have been chosen:

Revenue Maximisation The main motivation of most of the providers that enter in a Utility Computing Market is to earn money by selling its resources. Then it is important to consider a utility function to maximise the economic benefit.

Client Classification Some corporative clusters that most of the time have more resources than the amount that they actually need would like to sell their spare resources in a market for minimising the economic expenses of having its own resources. However, they only want to maximise its revenue to the external users and not to the users of internal or associated workgroups. The global utility function considered in this thesis allows the providers to perform Client Classification.

Minimise impact of peak hours Resource providers, especially web servers, have peak hours when their resources are overload and off-peak hours when the resources are almost idle. That means that a large amount of resources must be provisioned to avoid the collapse in peak hours, but this supposes a huge extra cost for more resources that will not be used in off-peak hours. Resource providers must find mechanisms to motivate clients for buying more resources at off-peak hours to minimize the workload of peak hours.

Maximisation of reputation When a provider violates a SLA due to a system failure or a bad calibration of provisioned resources, it must pay a penalty to the client, but also will loss its reputation in the market. In consequence, future clients will be reticent to buy these resources. There is needed a mechanism for compensating the problems derived of the loss of reputation.

Let $O \subseteq \{o_{rv}, o_{cc}, o_{ph}, o_{rp}\}$ the set of objectives, where:

- o_{rv} is the objective that defines the maximisation of the revenue. The higher is the revenue the higher is u_{rv}.
- o_{cc} is an objective used for client classification [46]. This gives preference to the local users (or users from a near organisation) over the users from non-related organisations.
- o_{ph} is the objective that gives preference to tasks or services to be executed in off-peak hours, to prevent the system overload during peak hours.
- o_{rp} is the objective used for maximising the reputation of the provider [27].

This paper demonstrates how the behaviour of the provider can be modulated only by changing the values of the components of O that multiply their associated sub-utility functions $u_{rv}, u_{cc}, u_{ph}, u_{rp}$ in negotiation time as can be shown in the general utility function (Equation 5.2).

$$U(S) = o_{rv}u_{rv} + o_{cc}u_{cc} + o_{ph}u_{ph} + o_{rp}u_{rp} \tag{5.2}$$

The next subsections describe and justify the sub-utility functions chosen in this paper, calculated in base to the SLA $S \subseteq \{M, C, CP, Rev, \Delta t\}$, where M, C are the Memory and CPUs amount to acquire, $0 \leq CP \leq 0$ is the indicator of Client Priority, Δt is the time slot where the resources are assigned and Rev is the revenue acquired by the sale. All the sub-utilities are normalised to the same range $[-1, 1]$ because otherwise the influence of the weights O would be distorted by the differences between the ranges of the sub-utilities.

Price Maximisation

Before describing u_{rv}, u_{cc}, u_{ph} and $u_{rp}0.$, it is advisable to describe an utility function that it is not used as a term of $U(S)$ but that some of the sub-utility functions depends on: the price maximisation utility function.

When the provider proposes a price, it must know the range of prices where the agreement is possible. Too much low prices are not profitable for the seller, and the client would not buy services at too high prices because it cost will be higher than the benefit of using them. When negotiating the sale of a good, the **reservation price of the seller** is the minimum price that the seller can accept without losing money for the sale. The **reservation price of the buyer** is the maximum price it can pay and do a sale which is beneficial for its objective (see figure 5.2). An agreement between buyer and seller is only possible when $RP_s \leq Rev \leq RP_b$.

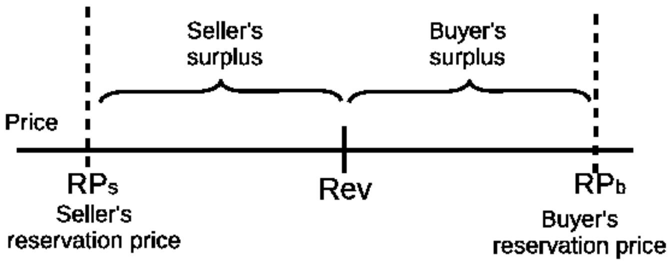

Figure 5.2: Reservation Price of both Buyer and Seller

Equation 5.3 defines the utility for given revenue:

$$u_p(S) = \frac{Rev - RP_s}{RP_b - RP_s} \tag{5.3}$$

That means that the utility of the price for the provider is higher (~ 1) when the Revenue that the provider obtains tends to be RP_b. However, this sub-utility function is not used directly as a term in $U(S)$, because in a competitive market high prices will enforce clients to look for cheaper providers, then the sale will not be performed and the benefit would be 0 (much less than a sale with a cheap price).

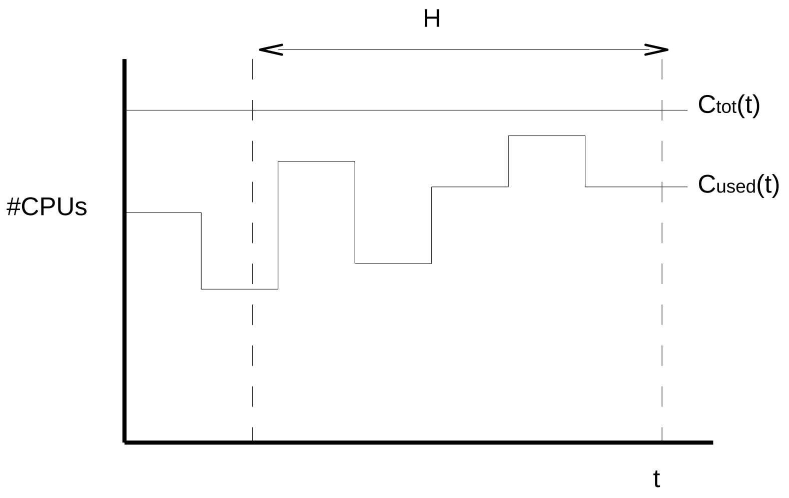

Figure 5.3: The aggressiveness factor is calculated in function to the current usage of the resources

The main issue of implementing this formula is to know the reservation price of the buyer, which only can be speculated in function of the market history.

Total Revenue Maximisation

For maximising the total revenue, it is needed to have into account the price of the sale, but also the status of the competitive market. Having into account the *Law of Supply and Demand* (see section 2.1), it is needed to define u_{rv} to propose different prices in function of the market status, so they will tend to be higher in demand excess scenarios, and lower in offer excess scenario.

To check the market status, an **aggressiveness factor** has been defined: an indicator of how aggressive must the pricing policy of the provider be. Its value is from 0 to 1 and is in function of the current aggressiveness value and the status of the resources.

Figure 5.3 shows the elements used to calculate the aggressiveness factor. Let t the current time, H the length of an historic time period, $C_{tot}(t)$ be a constant function whose value is the number of CPUs of the Provider, and $C_{used}(t)$ be a function that describes the number of busy CPUs in the provider over time. Equation 5.4 calculates the *ideal aggressiveness factor* $a'(t)$.

$$a'(t) = \frac{\int_{t-H}^{t} C_{used}(t)\,dt}{\int_{t-H}^{t} C_{tot}(t)\,dt} \tag{5.4}$$

Let $0 \leq \delta \leq 1$ the *aggressivenes adjustment rate* which shows how quick the actual aggressivenes will tend to the ideal aggressiveness, equation 5.5 shows how the aggressiveness $a(t)$ is adjusted in function of the ideal aggressiveness and the previous actual aggressiveness:

$$a(t) = a'(t)\delta + a(t-1)\,(1-\delta) \tag{5.5}$$

In the experiments performed in this paper, both CPU and Memory are negotiated. But since CPU is the bottleneck, it is used as a resource for calculating the

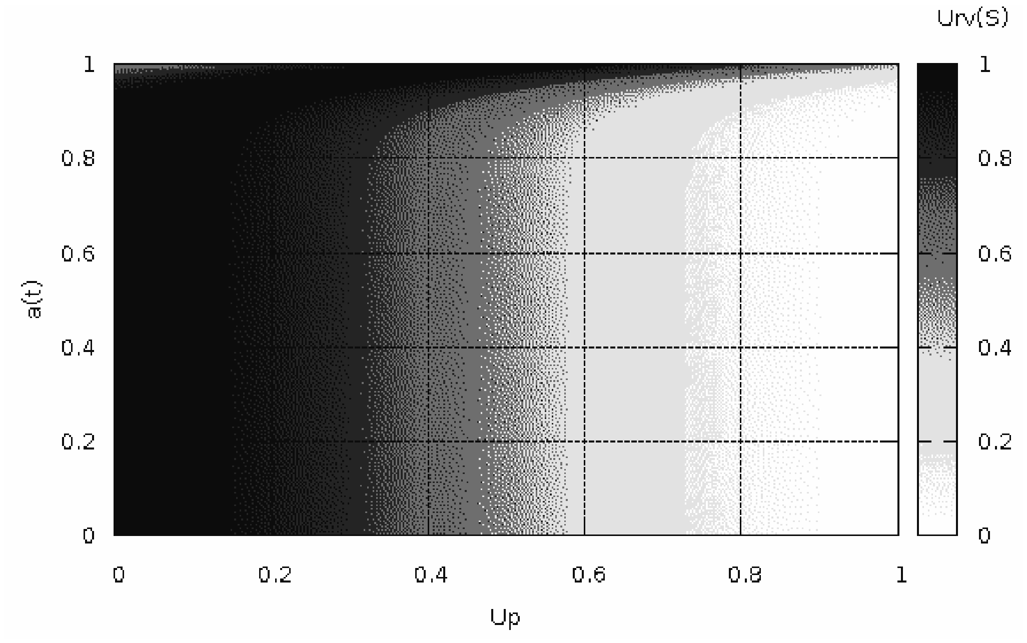

Figure 5.4: Colourmap that represents the value of $u_{rv}(S)$ in function of $u_p(S)$ and $a(t)$

peak hours.

Once $a(t)$ are defined $u_p(S)$, there is enough data to define an $u_{rv}(S)$ to achieve the next goals:

- In an offer excess scenario, where $a(t)$ is low, clients will choose providers offering lower prices ($u_p(S) \rightarrow 0$) for the same SLA. So $u_{rv}(S) \rightarrow 1$ when $u_p(S) \rightarrow 0$.
- In a demand excess scenario, where $a(t)$ is high, clients will have to accept high prices ($u_p(S) \rightarrow 1$), since they have very few alternatives. So it is convenient for the provider to push up its prices in order to maximize its benefit.

Having these points into account, and after several tests and simulations, the next utility function has been defined and the constant values have been tuned to provide the best utility in the most of the cases:

$$u_{rv}(S) = 0.5 + \frac{\sin\left(\frac{\pi}{2}\left(2u_p(S) + (1 - a(t)^{15})\right)\right)}{2} \tag{5.6}$$

The colour map in figure 5.4 helps to understand better the function in equation 5.6. The dark zones shows these combinations of $u_p(S)$ and $a(t)$ that gives higher values for u_{rv}.

It is convenient to explain how an intuitive and easy-to-understand initial function for $u_{rv}(S)$ has evolved to the so much complicated equation in 5.6. The first attempt for defining u_{rv} was the next function:

$$u_{rv}(S) = \sin\left(\frac{\pi}{2}\left(u_p(S) + (1 - a(t))\right)\right) \tag{5.7}$$

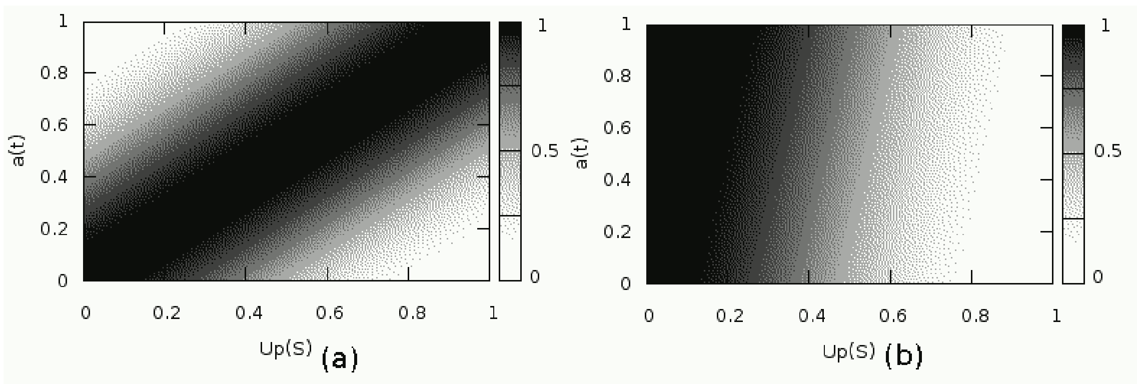

Figure 5.5: Previous intends for defining $u_{rv}(S)$

The intuition said that the Law of Demand and Offer could be accomplished by adjusting linearly the prices in function of the demand, as shown in the maximum values (darkest colour) of figure 5.5(a). However, the experimentation results shown that, even when a(t) is relatively high, the clients have some chances for choosing cheaper providers, so maximising $u_{rv}(S)$ would lead to have less revenue.

The second chance was to divide $a(t)$ in equation 5.7, whose visualization can be seen in figure 5.5(b). That function worked in normal market status, but not in those where the demand excess was extremely high ($a(t) \simeq 1$), because it did not take profit from the good position of the provider in the negotiation.

After some more tests, the best solution in all the market scenarios was to put the higher value of $u_{rv}(S)$ in $u_p(S) \simeq 0.5$ when $a(t) \to 1$, and let the maximum of $u_{rv}(S)$ near 0 in all the other cases. The result is the equation 5.6, used in the experiments performed for showing the results of this thesis.

Client Classification

In this paper, client classification is performed through price discrimination [46]. The parameter CP is the Client Priority, it tends to 1 when the Client is much related to the organisation of the provider, and tends to 0 when there is absolutely no relation between the Client and the Provider. It is calculated as the Euclidean distance between Client and Provider in a multi-dimensional space.

Equation 5.8 is used to define the utility for client classification. Given u_{rv} and CP, if the Client priority is high, the utility will be higher when u_{rv} is low (the provider must not be expensive for related clients). If the Client priority is low, the utility will be higher when u_{rv} is high (see figure 5.6).

$$u_{cc}(S) = \begin{cases} CP + u_p(S) & \text{if } u_p < 1 - CP \\ 2 - CP - u_p(S) & \text{otherwise} \end{cases} \tag{5.8}$$

Prioritisation of Off-Peak hours

Let $\Delta t = t_f - t_i$ be the interval of time where the task is executed, $C_{tot}(t)$ be a constant function whose value is the number of CPUs of the Provider, $C(t)$ be a constant function whose value the number of CPUs requested to the task under negotiation, and $C_{used}(t)$ be a function that describes the number of busy CPUs in

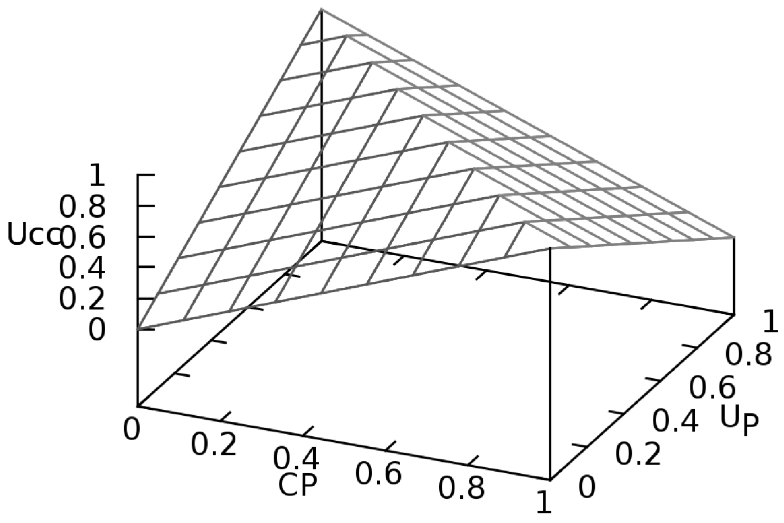

Figure 5.6: Utility function used for Client Classification

the provider over time. Equation 5.9 is the utility function that is higher when more resources are free, and near 0 when the provider resources are near its maximum occupation.

$$u_{ph}(S) = 1 - \frac{\int_{t_i}^{t_f} C_{used}(t) + C(t)\,dt}{\int_{t_i}^{t_f} C_{tot}(t)\,dt} \tag{5.9}$$

In the experiments, both CPU and Memory are negotiated. But since CPU is the bottleneck, it is used as a resource for calculating the peak hours.

Utility for reputation

Let R_0 be the reputation of the provider in negotiation time, R be the future reputation of the provider in case of SLA violation (updated as described in equation 5.11), and P the probability of violating an SLA (calculated in base to past statistical data); Equation 5.10 shows the utility of keeping the provider's reputation:

$$u_{rp}(S) = \frac{P\,R + (1-P)R_0}{R_0} - 1 \tag{5.10}$$

Notice that this utility function ranges from -1 to 0, since losing the current reputation is bad, and keeping it is neither good nor bad for the provider.

The value of R is calculated as described in [27]:

$$R = R_0(F_q + (1 - F_q)S) \tag{5.11}$$

, where S is the seriousness of the violation (a value between 0 and 1), and F_q is the factor for the reputation reduction in function to offered Quality of Service q. F_q can have values from 0 to 1: it will be greater for low-class providers and smaller for high-quality providers. This formulation assumes that providers are less allowed to have service failures for gold-class SLAs, and customers should be more tolerant with bronze-class SLAs. In the experiments, $F_{gold} = 0.5$, $F_{silver} = 0.75$ and $F_{bronze} = 0.85$.

5.3 Maximising the utility function

When the provider receives an offer, it must specify a price and a range of time (if the requested time is not fixed) to maximise the utility function.

Maximising nonlinear utility functions can be pretty complex specially when exists multiple variables. Choquet Integrals [44, 43, 47] have been used for multicriteria decision with nonadditive functions where some of their values are probabilistic, by using fuzzy logic. However, it does not help to maximise the function, but only choose the best alternative in a set.

Fortunately, the framework used in this thesis uses discrete values of time and price, and does not need fuzzy logic because the data used in the utility functions is well known by the provider (excepting the Reservation Price of the client, which is speculated). Then $U(S)$, which is theoretically continuum, is divided into a finite set of values in function of discretised price and time.

Choosing the best price and time slot is choosing the pair of price and time whose $U(S)$ is greater to the $U(S)$ values for all the other pairs. For example, in a negotiation where 3 time allocations are possible, $RP_b - RP_s = 10$ and the range of price is discretised in fractions of 0.1 currency units, 300 different values for $U(S)$ will be compared, and the pair of price and time who gives the highest $U(S)$ is chosen.

5.4 Why is the resource information needed?

This chapter has shown the importance of the usage of resource information for performing accurate negotiations. When calculating $u_p(S)$ for price maximisation, it is important to know the status of the resources and how an incoming SLA can affect into this status, in order to quantify them economically and calculate the Reservation Price of the Seller.

$u_p(S)$ and the aggressiveness factor $a(t)$ have a decisive role when calculating the utility for maximising the global revenue $u_{rv}(S)$. Section 5.2.2 shown how $a(t)$ is calculated in function to the historic monitoring data from the resources. The same historic data is also used to calculate $u_{ph}(S)$, that gives more importance to the jobs which are located in off-peak hours.

The resource information is also really important when calculating $u_{rp}(S)$, because the probability P of breaking an incoming SLA is calculated in function to statistical monitoring data of past executions and the current monitoring status.

Even an utility function such as $u_cc(S)$ used for performing Client Classification, has relation with the resource information, since it is calculated in function of $u_p(S)$.

Chapter 6

Experimentation Results

This chapter describes two experiments performed for checking the validity of the model. In the first experiment, a statistical method is used to check the influence of each of the parts of the model. Due to resources limitations (hundreds of different providers are compared in a same market), the model has been checked using a simulation of a market. The second experiment compares the effectiveness of using resource information in utility functions against providers that only implement fixed pricing (such as Amazon Elastic Compute Cloud[48]).

6.1 Simulation Environment

A simple market has been simulated to test the validity of the negotiation model. A Client Broker that can represent a Web Client or a Grid Client enters in the market to ask for web workload or for plain resources. The workload for grid has a pseudo-random distribution and the workload for web services follows a distribution as shown in figure 6.1, taken from a real web application, with peak and off-peak hours.

Grid Clients send a SLA proposal where is specified the plain resources (CPU and Memory) to buy, the duration of the job, and a time interval where the job can

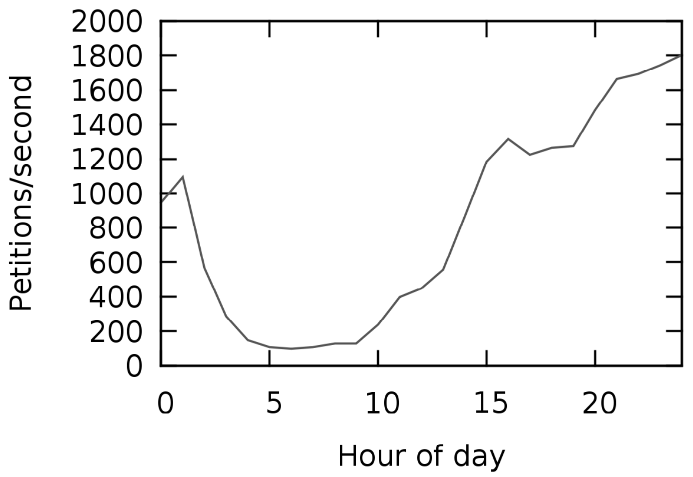

Figure 6.1: Sample of Daily workload of a Web Server

be executed (bigger than the duration, to let the EERM schedule the best execution time). Web Service Clients send a required workload for a service, and a fixed time interval to use the services (there is no arbitrary schedule of the reservation, since web users want the services for the same moment).

Both client types also must specify what QoS class they want: gold, silver or bronze. Gold clients will pay the triple than bronze clients, and silver clients the double than bronze ones. The average failure rate for gold, silver and bronze services are, respectively, 0.5%, 1% and 2%.

A Client looks for potential providers in the market discovery and sends SLA Proposals to all of them. After that, the providers accept/deny the proposals and return to it a time allocation and a price, based on the maximisation of their utility function. Finally, the Client chooses the Provider with a best price and time schedule for its interests and sends it a confirmation.

The provider can violate the SLA due to an internal error, or because it receives a proposal from another Client that can not be allocated but is interesting to accept it and cancel the other (it is decided by the utility function having into account objectives such as client classification or revenue maximisation) [25, 20] . This violation will affect to the reputation of the provider, which is taken into account by the Client in negotiation time: when choosing the best SLA, the price proposed by the provider is divided by its reputation, so the client will consider the price of a provider with low reputation higher than the same price from a provider with high reputation.

6.2 Checking statistically the effectiveness of the sub-utility functions

In this test, repeated simulations are performed in a competitive market with 100 providers, **whose objective weights of the utility function are varied and generated randomly**, to provide some statistically valuable data. Simulations are repeated with a number of clients that vary from 50 (offer excess) to 1000 (demand excess). Each provider is selling 20 CPUs and 6GB of RAM memory.

This section shows the results of the simulation in terms of the four objectives described in section 5.2.2. For each simulation, the next data is collected from the provider side:

Revenue Total Revenue of the provider.

AvgPrice Average price of Resource/Hour sold.

AvgAffinity Average affinity of clients that use the system (value CP of section 5.2.2).

AvgReputation Average reputation of the provider during the whole execution.

AvgOfferPrice The price of Resource/Hour that the Provider offers to the Client. The difference with *AvgPrice* is that it includes only the prices of the agreed negotiations, and *AvgOfferPrice* includes the offer prices for both the agreed and non-agreed negotiations.

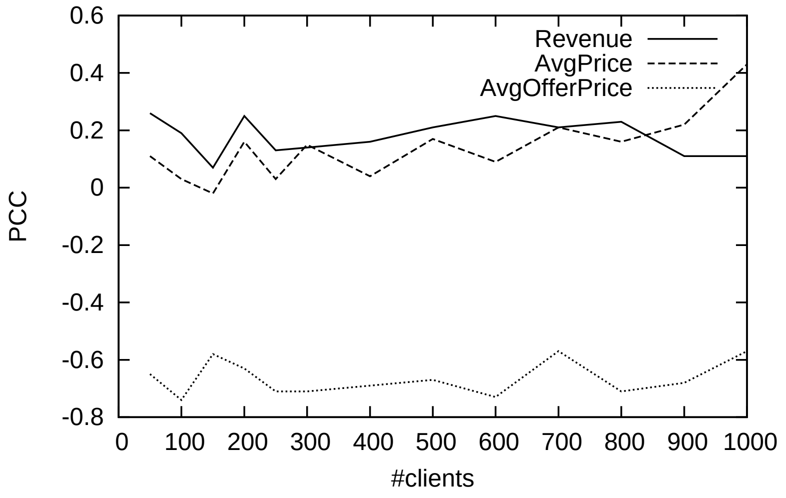

Figure 6.2: Correlation between o_{rv} and some output parameters

To show the effectiveness of the utility functions proposed in this thesis, the Pearson Correlation Coefficient (PCC) between the collected data and the objectives o_{rv}, o_{cc}, o_{ph} and o_{rp} is calculated. Equation 6.1 shows how to calculate the PCC between two sets of data X and Y. PCC calculates the relation between X and Y data sets, each one with N elements. Its value is in the range from +1 (perfect linear relationship) to -1 (perfect negative linear relationship). $PCC = 0$ means that there is no linear relationship.

$$PCC = \frac{\sum XY - \frac{\sum X \sum Y}{N}}{\sqrt{\left(\sum X^2 - \frac{(\sum X)^2}{N}\right)\left(\sum Y^2 - \frac{(\sum Y)^2}{N}\right)}} \tag{6.1}$$

6.2.1 Revenue Maximisation

Figure 6.2 shows how o_{rv} has a slightly influence in the total revenue of the provider (around 0.2). Obviously the correlation coefficient can not be 1 because there are many other factors that have influence in the revenue. However, there is a big negative linear relationship between o_{rv} and the price that the provider proposes for the sale of the resource in negotiation time (AvgOfferPrice): providers that want to sell more must decrease their prices. However it can be observed that o_{rv} has a positive influence on the prices of the sold resources (AvgPrice). It is because the maximisation of $U(S)$ will lead to ask the optimum prices in function of the market status (speculated by the aggressiveness function $a(t)$).

In extreme demand Excess scenario (900-1000 clients), where the provider can be more aggressive in its negotiations (because there are more clients interested on acquiring its services), it can be observed a positive correlation between o_{rv}, AvgOfferPrice and AvgPrice. However the correlation with the total revenue is more or less the same. That does not mean that the utility function is less efficient, it means that all the providers increase their revenues because the market status,

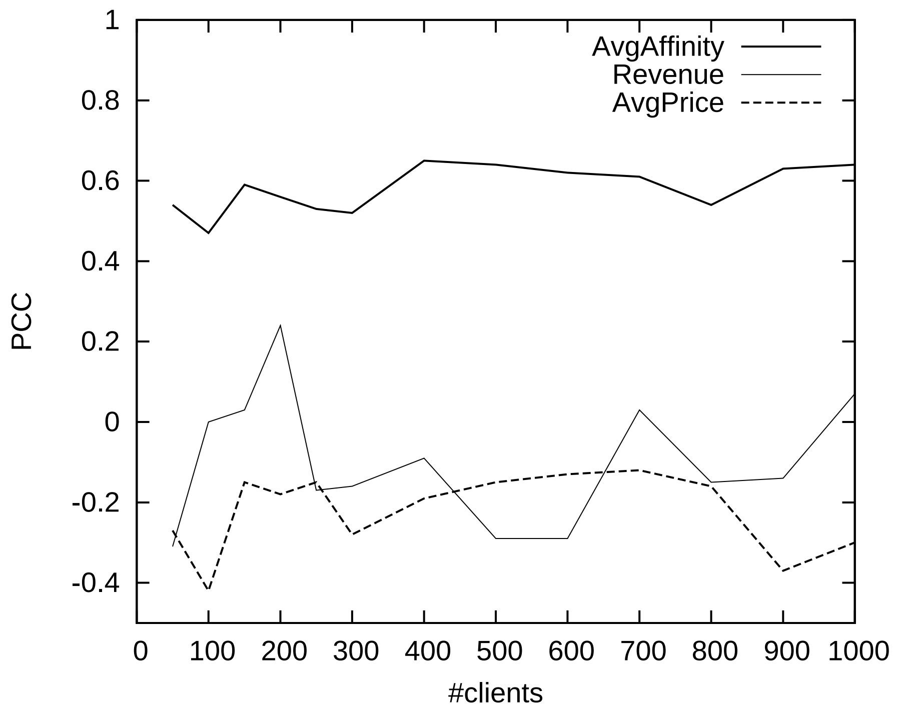

Figure 6.3: Correlation between o_{cc} and some output parameters

and the influence of o_{rv} in the total revenue is less in percentage.

6.2.2 Client Classification

Figure 6.3 shows the effectiveness of the inclusion of $o_{cc}u_{cc}(S)$ in the general utility function: the higher is o_{cc}, the higher is the affinity (around 0.6 in all the market scenarios).

As described in section 5.2.2, u_{cc} is strongly related with u_p. Figure 6.3 reflects this relation as a negative correlation between the o_{cc}, the global revenue, and the average price. Since the provider will try to sell to affine customers, it will offer them its resources at lower prices, and there are more possibilities that clients choose affine providers.

6.2.3 Priorisation of off-peak hours

Workload for Web Services have fixed intervals but an irregular distribution (figure 6.1), and workload for Grid Tasks have a random distribution, but since they are not real-time applications, they can be scheduled to be executed in the future. Figure 6.4 shows how the inclusion of $o_{ph}u_{ph}(S)$ in $U(S)$ makes the providers the possibility for giving better prices to the clients in off-peak hours and, in consequence, the Grid Jobs are automatically executed when the Web Services workload is low.

6.2.4 Reputation

Figure 6.5 shows that, unlike we expected before running the simulations, o_{rp} does not have any influence on the average reputation of the provider. However, the

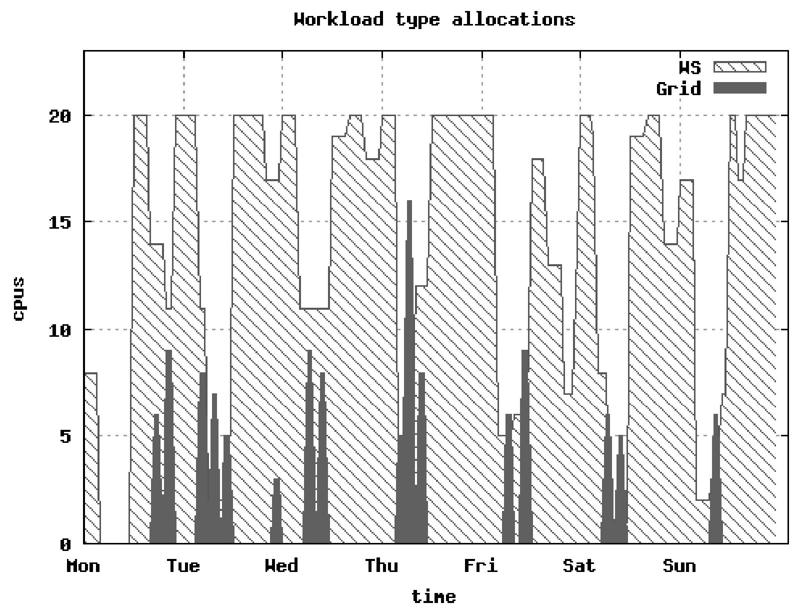

Figure 6.4: Allocation in time of workloads divided by Web Services or Grid

results are interesting because u_{rp} acts as a risk manager. The figure show how the average price of the sold resources is increased or decreased in function to the reputation. This means that the provider will charge a small amount of money to compensate possible losses as consequence of the loss of reputation.

Figure 6.6 shows clearly the importance of keeping a high reputation. In the experiments, the revenue increases almost linearly with the reputation. At equal prices, a Client will choose the Provider with higher reputation. The alternative to providers with low reputation is to decrease their prices.

6.3 Comparison with fixed-pricing providers

In this experiment, 10 different providers have been compared: there are four providers that implement negotiation as proposed in this thesis and six providers that implement fixed pricing. Each one of the four providers that use non-additive utility maximisation for negotiation has a main objective whose o weight value is 0.55 and the other secondary objectives have a weight of 0.15. So there is a provider that prioritises revenue maximisation, other that prioritises client classification, one that prioritises off-peak hours and other that prioritises reputation maximisation. On the side of providers with fixed pricing, since it is difficult to know beforehand what is the best fixed price; six providers with different prices have been added into the testbed. Each provider proposes always a fixed percentage between the Provider's Resource Price and the Buyer's Resource Price, by having fixed values of $\alpha = \{0.04, 0.06, 0.08, 0.10, 0.12, 0.14\}$ in the next pricing formula:

$$Price = RP_s + \alpha \left(RP_b - RP_s \right)$$

Since fixed-pricing providers do not have influence, in a competing market, in the peak minimisation nor the reputation maximisation decisions of their competitors,

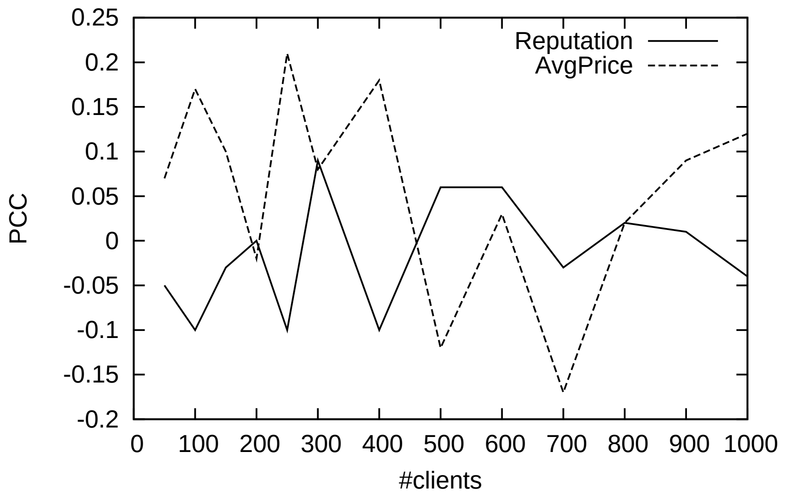

Figure 6.5: Correlation between o_{rp} and some output parameters

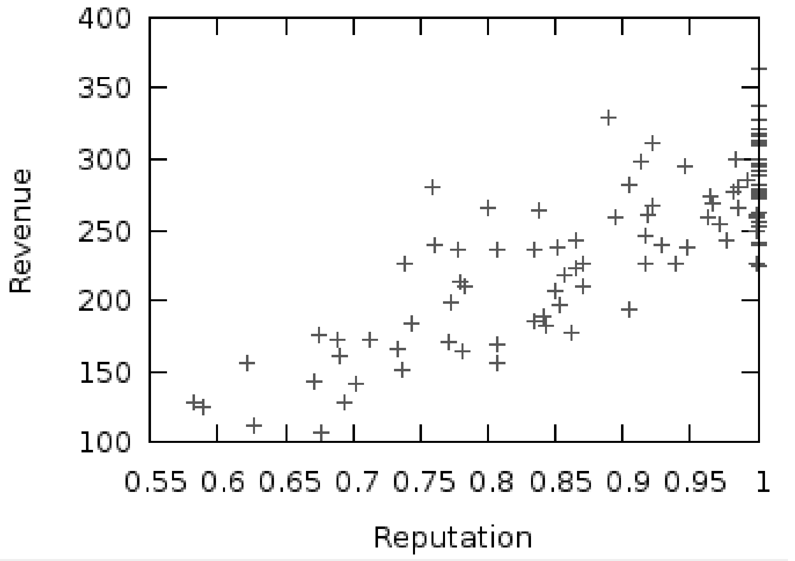

Figure 6.6: Relation between reputation and revenue

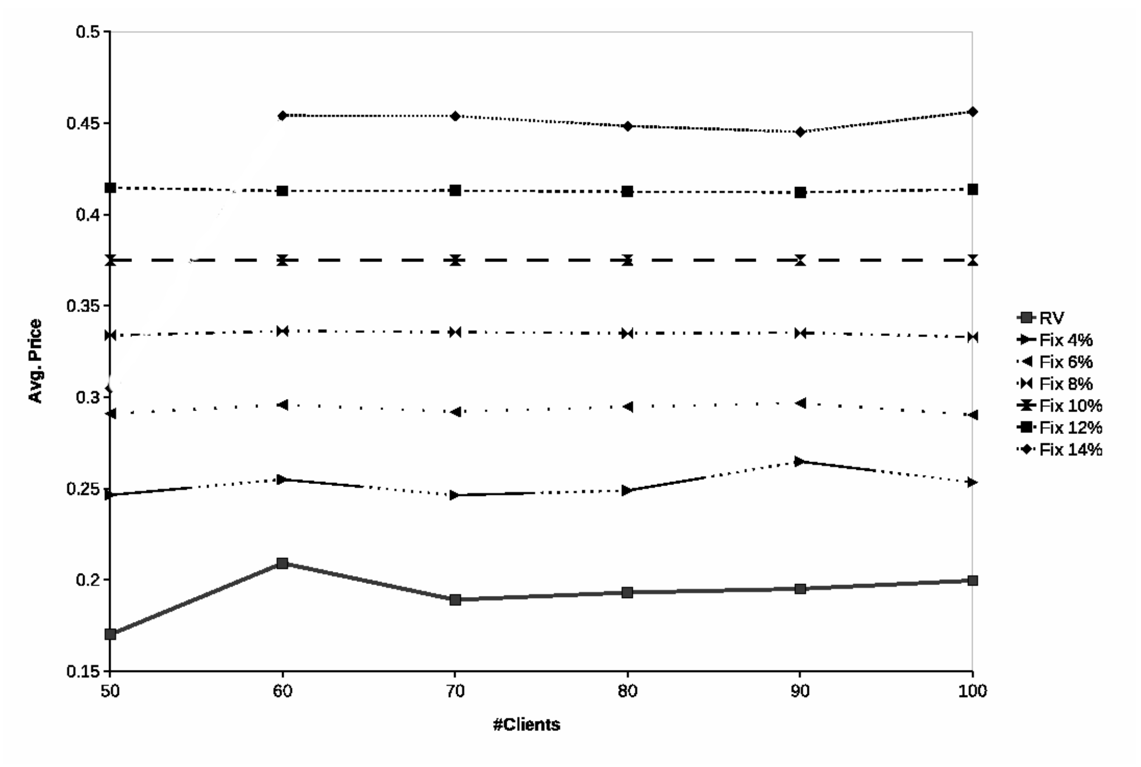

Figure 6.7: Comparison of average sales price between a provider that tries to maximise the revenue (RV) with fixed-pricing providers

only providers that maximise revenue and perform the client classification have been compared with the fixed-pricing providers. Furthermore, the influence of not implementing peaks minimisation in them is shown. However, the influence of policies for reputation maximisation can not be shown, because the data set in this experiment is too small to establish correlations and, as shown in subsection 6.2.4, the influence of $o_{rp}u_{rp}(S)$ can not be compared in terms of achieved reputation with the other providers.

6.3.1 Revenue maximisation

Figure 6.7 shows how the adaptive pricing scheme of the provider that maximises revenue (RV) always is lower than the fixed prices of its competitors. Only the data of offer excess scenarios (50 to 100 clients) are shown, because with fewer clients, some fixed pricing providers, specially those with higher prices, do not sell enough resources, so the average price can not be measured. A proof of that fact is reflected when there are only 50 clients, in the provider that sells the resources at fixed price that is the 14% between the Provider's Resource Price and the Buyer's Resource Price (annotated as Fix 14% in the graph). Since it is the higher price, it does not sell any resource.

Figure 6.8 shows that adaptive pricing by maximisation of nonadditive utility functions is the best choice in almost all the scenarios. In the high excess of offer scenarios (20 clients), fixed pricing providers that sells its resources at low price (Fix 4% and Fix 6%) have more revenue than adaptive pricing providers (RV). This is

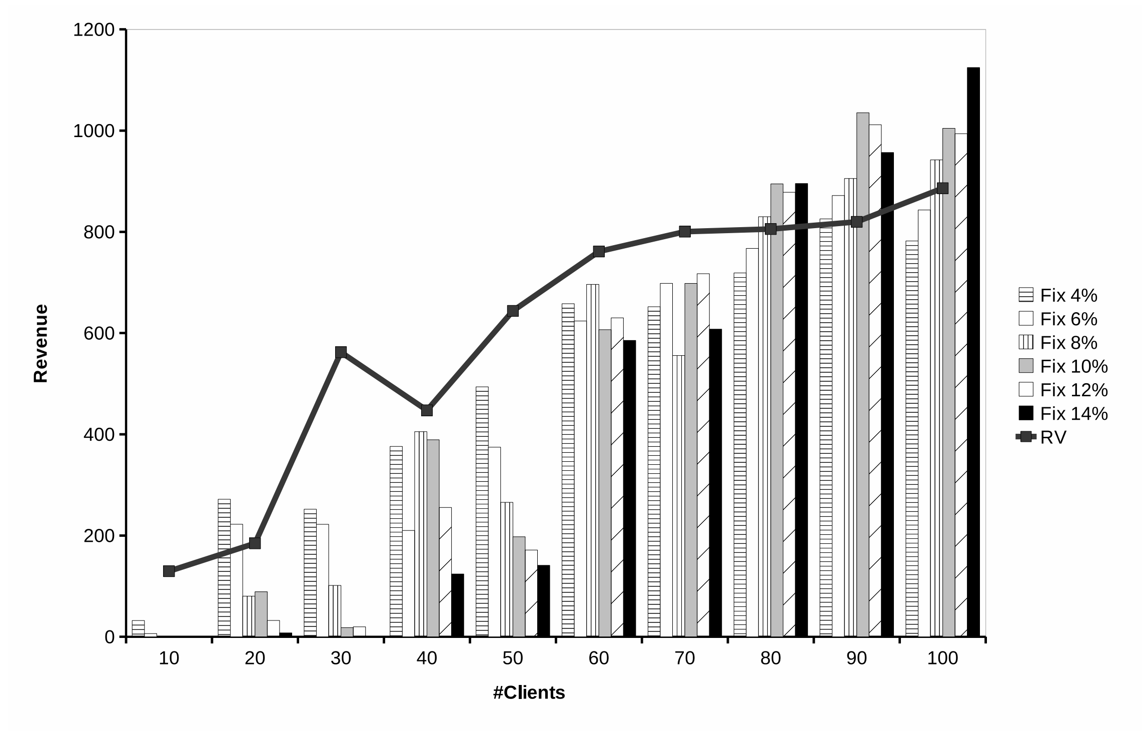

Figure 6.8: Comparison of revenue between a provider that tries to maximise the revenue (RV) with fixed-pricing providers

because the biggest part of the demand is shared across providers with low prices, and since these providers sell its resources at higher prices than RV, they earn more money.

Also in the highest demand scenarios (8 to 10 clients per each provider), the fixed pricing providers with highest prices earn more money than the adaptive pricing one. In this case, clients do not have enough alternatives for choosing and they must accept almost any offer. Providers with highest prices can take advantage of this situation.

Taking into account results of figure 6.8, the question that arises is: why do not change u_{rv} for increasing its prices in the lowest and highest demand scenarios? The response is that these scenarios are a unreal, since markets tend to equilibrium status (where the revenue of RV is higher than the revenue of competitors) and, as observed in the experiments performed when defining u_{rv}, when all the providers use adaptive pricing, increasing the prices in these situations can lead to a decreasement of global revenue.

6.3.2 Client Classification

Figure 6.9 shows in a graphical way how efficient is client classification compared to providers that do not perform it (the fixed pricing ones).

It can be seen how average affinity of clients decreases when the clients number increases. It is because the provider with client classification can accept almost all the affine users when its resources are idle, but when it is overloaded, most of the clients can not use its resources and must look for less affine ones. A way of

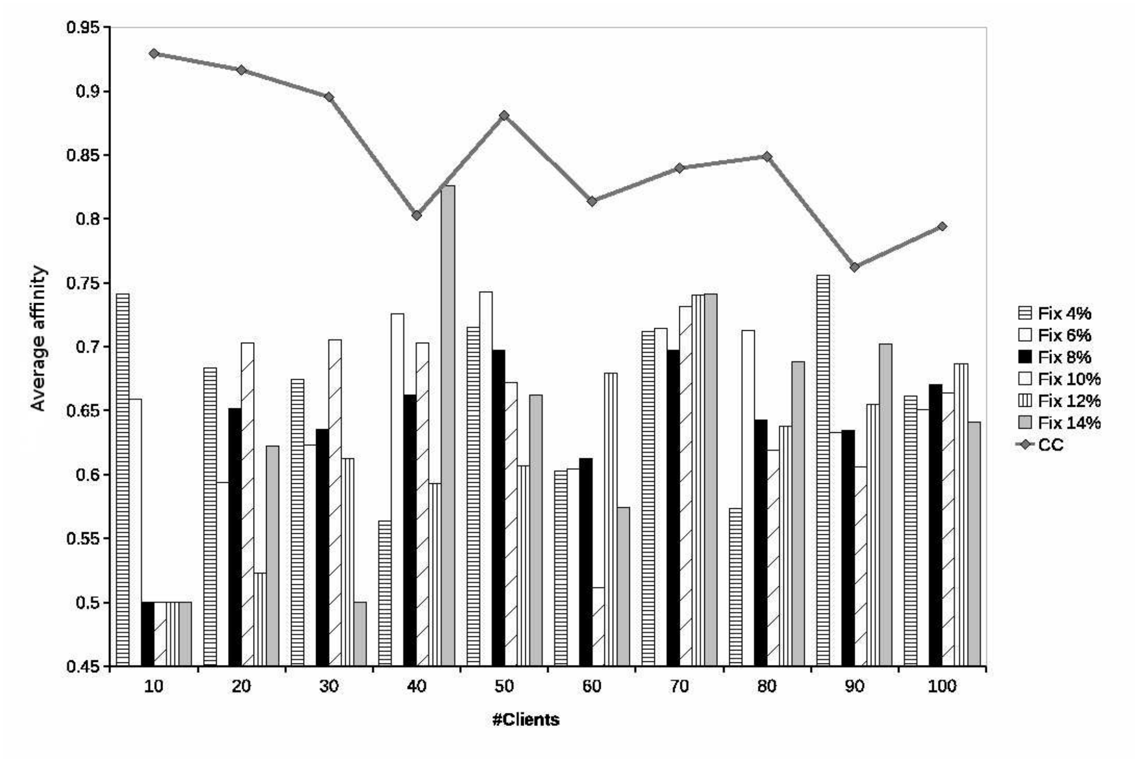

Figure 6.9: Comparison of client affinity between a provider that performs client classification (CC) with fixed-pricing providers

keeping high affinity in all the scenarios could be by implementing job cancellation for low-affinity users. But this solution will entail other important problems, such as economic losses due to the pay of penalties and the loss of reputation.

6.3.3 Peak-hours Minimisation

Figure 6.10 shows clearly the influence of not having policies for the minimisation of off-peak hours. Since the providers that use non-additive utility functions maximisation can allocate the workloads in off-peak hours at better prices (as shown in figure 6.4), providers that do not implement peak minimisation policies do not execute Grid Workloads. In the experiments performed, only the provider that offered the lowest fixed prices (4% over RP_s) in demand excess scenario executed 3 Grid jobs. The allocations for all the other providers are similar to figure 6.10.

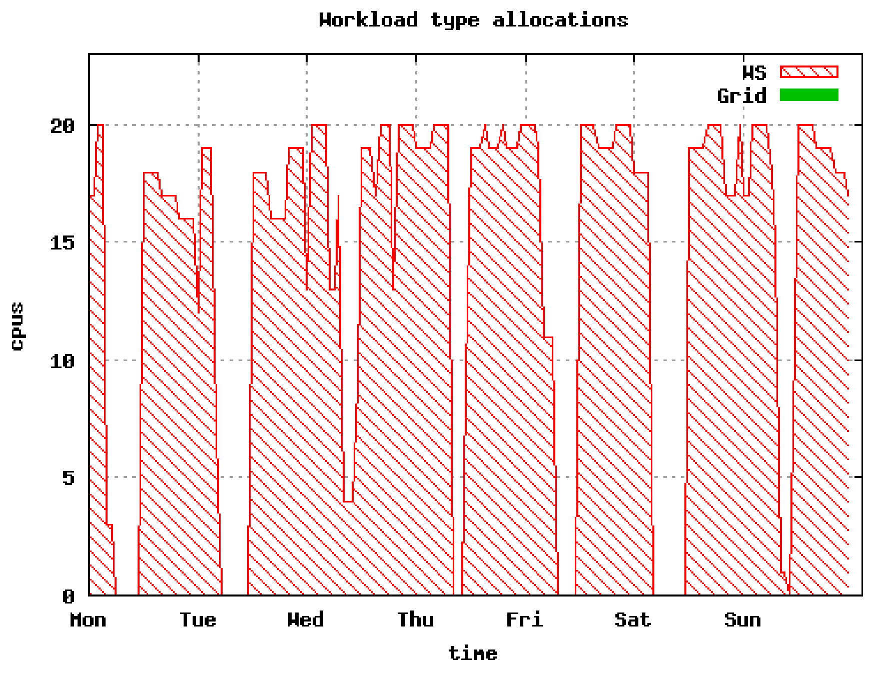

Figure 6.10: Sample distribution of workload types in a provider without peak-hours minimisation policies. It can be observed that there are no Grid workloads

Chapter 7

Conclusions and future work

The first aim of this thesis is to show the benefits of applying knowledge about economics to resource management and *vice-versa*. In Utility Computing Markets, having knowledge about economic status of the market will help to the resource manager to perform its work not considering only technical metrics such as performance, but also economic goals, such as client classification, revenue maximisation or reputation maximisation. In the opposite direction, having knowledge about the status of the traded resources will lead to the market broker to perform better economic decisions. This thesis demonstrated how the figure of an Economically Enhanced Resource Manager can benefit to resource providers by providing resource data to the brokers and by considering economic policies in resource management.

The other contribution of this thesis is the intention of being a step forward in the modelling and evaluation of utility functions for negotiations in utility computing markets. The simulations show how a provider can perform complex actions by only maximising a multi-dimensional utility function. The contribution of these experiments lies in the usage of non-additive utility functions, more difficult to treat, but needed when assuming that the terms under negotiation are not independent between them. In the model defined in chapter 5, the utilities for client classification $u_{cc}(S)$ and revenue maximisation $u_{rv}(S)$ were related by the price: maximising the price would lead to maximise $u_{rv}(S)$, but to minimise $u_{cc}(S)$ for affine clients.

The proposed non-additive utility function considers the possibility of having multiple objectives in a same entity, such as revenue maximisation, client classification, reputation or load-balancing in time. In order to keep the efficiency both in economic and performance terms, most of the parameters that compose the utility function are collected dynamically from the resource-level information. This thesis has shown the high importance of having this information available in negotiation time. The simulations performed demonstrate how the objectives can be partially achieved by balancing correctly their weights in the utility function.

7.1 Future Work

This thesis leaves some open lines for future research:

- Improve utility functions for more efficient negotiations and extend its terms to include other economic or performance goals.

- Find methods for the maximisation of complex non-additive utility functions that include fuzzy values for nondeterministic data.
- Evaluate the validity of the model in a real Utility Computing market, taking real data from the resource fabrics and compare it with other existing models.

7.2 Related publications

Part of the work performed in this thesis has been published in several articles and papers.

The need for applying economic enhancements to resource management was considered on reference [21], and an example of client classification as economic policy applied to resource management was described in [46].

Reference [25] described a model for maximising the revenue of a service provider by combining resource reallocations and selective SLA violations. The validity of the model was demonstrated in a journal article that extended the previous work [20].

The implementation of the utility function for reputation maximisation and its influence in the negotiation is based in the work performed in the paper [27].

The results of the preliminary experiments of this thesis have been submitted and will be published in conference proceedings [49]. In addition, the work of this thesis is being extended and their results will be described in another paper.

Bibliography

[1] J. Basney and M. Livny, “Deploying a high throughput computing cluster,” in *High Performance Cluster Computing: Architectures and Systems, Volume 1*, R. Buyya, Ed. Prentice Hall PTR, 1999.

[2] M. A. Rappa, “The utility business model and the future of computing services,” *IBM Syst. J.*, vol. 43, no. 1, pp. 32–42, 2004.

[3] D. Thain and M. Livny, “Building reliable clients and servers,” in *The Grid: Blueprint for a New Computing Infrastructure*, I. Foster and C. Kesselman, Eds. Morgan Kaufmann, 2003.

[4] R. Buyya, C. S. Yeo, and S. Venugopal, “Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities,” *High Performance Computing and Communications, 10th IEEE International Conference on*, vol. 0, pp. 5–13, 2008.

[5] S. Conway, R. Walsh, and E. C. Joseph, “Hpc management software: reducing the complexity of hpc cluster and grid resources,” Platform Computing Co., Tech. Rep., 2008.

[6] I. Goiri, “Towards virtualized service providers,” Master’s thesis, Technical University of Catalonia, 2008.

[7] A. V. Moorsel, “Metrics for the internet age: Quality of experience and quality of business,” 5th Performability Workshop, Tech. Rep., 2001.

[8] D. F. Ferguson, C. Nikolaou, J. Sairamesh, and Y. Yemini, “Economic models for allocating resources in computer systems,” in *Market Based Control of Distributed Systems. World Scientific.* Press, 1996, pp. 156–183.

[9] J. V. Neumann and O. Morgenstern, *Theory of Games and Economic Behavior.* Princeton University Press, 1944. [Online]. Available: http://jmvidal.cse.sc.edu/library/neumann44a.pdf

[10] Web Services Agreement specification (WS-Agreement). [Online]. Available: www.ogf.org/documents/GFD.107.pdf

[11] H. Raiffa, *The art and science of negotiation.* Cambridge, Mass: Belknap Press of Harvard University Press, 1982.

[12] P. Faratin, C. Sierra, and N. R. Jennings, "Negotiation decision functions for autonomous agents," *International Journal of Robotics and Autonomous Systems*, vol. 24, pp. 3–4, 1998.

[13] L. Zadeh, K. Fu, K. Tanaka, and M. Shimura, *Fuzzy sets and their applications to Cognitive and Decision Processes.* Academic Press, New York, 1975.

[14] D. Ouelhadj, J. Garibaldi, J. MacLaren, R. Sakellariou, and K. Krishnakumar, "A multi-agent infrastructure and a service level agreement negotiation protocol for robust scheduling in grid computing," in *Proceedings of the European Grid Conference*, ser. Lecture Notes in Computer Science. Springer-Verlag, 2005.

[15] R. G. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *Transactions on Computers*, vol. C-29, no. 12, pp. 1104–1113, 1980. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1675516

[16] N. Vulkan and N. R. Jennings, "Efficient mechanisms for the supply of services in multi-agent environments," in *ICE '98: Proceedings of the first international conference on Information and computation economies.* New York, NY, USA: ACM, 1998, pp. 1–10.

[17] J. Altmann, C. Courcoubetis, G. D. Stamoulis, M. Dramitinos, T. Rayna, M. Risch, and C. Bannink, "Gridecon: A market place for computing resources," in *GECON*, 2008, pp. 185–196.

[18] Self-organizing ICT Resource Management (SORMA). [Online]. Available: http://www.sorma-project.eu

[19] Information Society Technologies Programme. [Online]. Available: http://www.cordis.lu/ist

[20] M. Macías, O. Rana, G. Smith, J. Guitart, and J. Torres, "Maximizing revenue in Grid markets using an Economically Enhanced Resource Manager," *Concurrency and Computation: Practice and Experience*, vol. 9999, no. 9999, p. n/a, September 2008. [Online]. Available: http://dx.doi.org/10.1002/cpe.1370

[21] T. Püschel, N. Borissov, M. Macías, D. Neumann, J. Guitart, and J. Torres, "Economically enhanced resource management for internet service utilities." in *WISE*, ser. Lecture Notes in Computer Science, B. Benatallah, F. Casati, D. Georgakopoulos, C. Bartolini, W. Sadiq, and C. Godart, Eds., vol. 4831. Springer, 2007, pp. 335–348. [Online]. Available: http://dblp.uni-trier.de/db/conf/wise/wise2007.html

[22] D. E. Campbell, *Resource Allocation Mechanisms.* Cambridge University Press, London, 1987.

[23] P. R. Wurman, "Market structure and multidimensional auction design for computational economies," Ph.D. dissertation, 1999, chair-Michael P. Wellman.

[24] R. Nou, F. Julià, and J. Torres, "Should the grid middleware look to self-managing capabilities?" 2007.

[25] M. Macías, G. Smith, O. Rana, J. Guitart, and J. Torres, "Enforcing service level agreements using an economically enhanced resource manager," in *Proceedings of hte 1st Workshop on Economic Models and Algorithms for Grid Systems (EMAGS 2007)*, Austin, Texas, USA, September 2007.

[26] C. S. Yeo and R. Buyya, "Pricing for utility-driven resource management and allocation in clusters," *International Journal of High Performance Computing Applications*, 2007.

[27] M. Macías and J. Guitart, "Influence of reputation in revenue of grid service providers," in *Proceedings of the 2nd International Workshop on High Performance Grid Middleware (HiPerGRID 2008)*, Bucharest, Romania, November 2008.

[28] S. Newhouse, J. MacLaren, and K. Keahey, "Trading grid services within the uk e-science grid," pp. 479–490, 2004.

[29] K. Djemame, I. Gourlay, J. Padgett, G. Birkenheuer, M. Hovestadt, O. Kao, and K. Voß, "Introducing risk management into the grid," *e-science*, vol. 0, p. 28, 2006.

[30] Grid 4 All. [Online]. Available: http://www.grid4all.eu

[31] F. A. Hayek, W. Bartley, P. Klein, and B. Caldwell, *The collected works of F. A. Hayek.* University of Chicago Press, 1989.

[32] I. T. Foster, "Globus toolkit version 4: Software for service-oriented systems." in *NPC*, ser. Lecture Notes in Computer Science, H. Jin, D. A. Reed, and W. Jiang, Eds., vol. 3779. Springer, 2005, pp. 2–13.

[33] M. Romberg, "Unicore: Beyond web-based job-submission," in *Proceedings of the 42nd Cray User Group Conference*, 2000, pp. 22–26.

[34] R. Buyya, "Economic-based distributed resource management and scheduling for grid computing," *CoRR*, vol. cs.DC/0204048, 2002.

[35] J. Guitart, M. Macías, O. Rana, P. Wieder, R. Yahyapour, and W. Ziegler, *Market-Oriented Grid and Utility Computing.* Wiley, 2009, no. 12, ch. SLA-based Resource Management and Allocation.

[36] C. S. Yeo and R. Buyya, "A taxonomy of market-based resource management systems for utility-driven cluster computing," *Softw. Pract. Exper.*, vol. 36, no. 13, pp. 1381–1419, 2006.

[37] M. J. Freedman, C. Aperjis, and R. Johari, "Prices are right: Managing resources and incentives in peer-assisted content distribution," in *Proc. 7th International Workshop on Peer-to-Peer Systems (IPTPS08)*, Tampa Bay, FL, Feb. 2008.

[38] A. Auyoung, B. N. Chun, C. Ng, D. Parkes, A. Vahdat, and A. C. Snoeren, "Practical market-based resource allocation," Tech. Rep., 2007.

[39] Y. Chen, S. Iyer, X. Liu, D. Milojicic, and A. Sahai, "Sla decomposition: Translating service level objectives to system level thresholds," *Autonomic Computing, 2007. ICAC '07. Fourth International Conference on*, pp. 3–3, June 2007.

[40] S. Kounev, R. Nou, and J. Torres, "Using qpn to add qos to grid middleware," Universitat Politècnica de Catalunya, Tech. Rep., 2007.

[41] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "Swrl: A semantic web rule language combining owl and ruleml," W3C Member submission 21 may 2004, Tech. Rep., 2004. [Online]. Available: http://www.w3.org/Submission/SWRL/

[42] G. D. Stamoulis, D. Kalopsikakis, A. Kyrikoglou, and C. Courcoubetis, "Efficient agent-based negotiation for telecommunications services," in *In Proceedings of Globecom '99, Rio de Janeiro*, 1999, pp. 1989–1996.

[43] S. Angilella, S. Greco, F. Lamantia, and B. Matarazzo, "Assessing non-additive utility for multicriteria decision aid," *European Journal of Operational Research*, vol. 158, pp. 734–744, noviembre 2004.

[44] T. Murofushi and M. Sugeno, "An interpretation of fuzzy measures and the choquet integral as an integral with respect to a fuzzy measure," *Fuzzy Sets Syst.*, vol. 29, no. 2, pp. 201–227, 1989.

[45] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," pp. 696–699, 1988.

[46] T. Püschel, N. borissov, D. Neumann, M. Macías, J. Guitart, and J. Torres, "Extended resource management using client classification and economic enhancements," in *Proceedings of eChallenges e-2007 Conference*, The Hague, Netherlands, October 2007.

[47] F. E. Baf, T. Bouwmans, and B. Vachon, "Foreground detection using the choquet integral," *Image Analysis for Multimedia Interactive Services, International Workshop on*, vol. 0, pp. 187–190, 2008.

[48] Amazon elastic compute cloud. [Online]. Available: http://aws.amazon.com/ec2/

[49] M. Macías and J. Guitart, "A non-additive negotiation model for utility computing markets," in *Proceedings of Jornadas del Paralelismo 2009, A Coruña, Spain*, 2009.