Simulation Tool for Market-Based Cloud Resource Allocation

Carlos Mera Gómez a, b
Supervisor: Dr. Rami Bahsoon b, c

MSc in Internet Software Systems

a School of Computer Science, University of Birmingham, Birmingham, United Kingdom
b cjm075@cs.bham.ac.uk
c r.bahsoon@cs.bham.ac.uk

Word count (excluding keywords, acknowledgments, table of content and bibliography): 13350

September 2011
Abstract

Market-Based Systems are aimed at improving the resource allocation and control in dynamic distributed systems by creating autonomous agents that carry out their trading based on market models and financial strategies. Cloud Computing is a model which offers IT infrastructure and applications as services in accordance with expected Quality of Services (QoS) parameters which are specified in a Service Level Agreement (SLA). The aim of this research is to design and explore a new market-based model which uses reputation mechanisms to minimize SLA violations in market-based Cloud architectures. In order to experiment with the impact of different reputation metrics, a simulation tool has been developed by extending a Cloud Computing framework called CloudSim. Two different market mechanisms, Posted Offer and Reverse Auction, have been chosen to compare and contrast their performance at implementing the resource allocation in the Cloud under varying resource configurations and experimental economics scenarios. The evaluation of the results in the performed simulations of this work shows that the reputation metrics significantly improve the resource allocation in distributed autonomous trading such as auctions.

Keywords: SLA, Market-Based Systems, Reputation Mechanism, Cloud Computing architectures, Experimental Economics.
Acknowledgements

I would like to thank Dr. Rami Bahsoon for sharing his ideas, giving me his guidance, encouraging me to carry out the research and keeping my confidence at the appropriate level in the most challenging activities to push the project ahead. Additionally, I would like to dedicate this effort to my father Carlos Mera, my mother Neris Gómez and my girlfriend Rommy Márquez for all their patience and support during my postgraduate studies.
Table of Content

Abstract ................................................................................................................................. I

Acknowledgements .............................................................................................................. II

Table of Content ................................................................................................................... III

1 Introduction ...................................................................................................................... 1

2 Background and Related Work ....................................................................................... 3
   2.1 Cloud Computing ........................................................................................................ 3
   2.2 Market-Based Systems ............................................................................................... 4
   2.3 Market-Based Cloud ................................................................................................... 5
   2.4 Experimental Economics ........................................................................................ 6
      2.4.1 Auctions .............................................................................................................. 7
      2.4.2 Posted Offer Auction ......................................................................................... 8
      2.4.3 Reverse Auction ............................................................................................... 8
      2.4.4 Bargaining ........................................................................................................ 9
      2.4.5 Double Auction ............................................................................................... 9
   2.5 Reputation Systems ................................................................................................... 10
   2.6 SLA in Cloud Architectures ...................................................................................... 11
   2.7 CloudSim .................................................................................................................. 12
      2.7.1 Simulation Approach ....................................................................................... 12
      2.7.2 CloudSim Architecture .................................................................................. 12
   2.8 Related Work .......................................................................................................... 14

3 Motivation ......................................................................................................................... 14

4 Research Objectives ...................................................................................................... 15

5 Challenges ....................................................................................................................... 15

6 System Design ................................................................................................................. 15
   6.1 Model Description .................................................................................................... 15
   6.2 Model Assumptions ............................................................................................... 17
   6.3 Trading with Reputation Metrics ............................................................................ 17
   6.4 Simulation Tool Design .......................................................................................... 19
      6.4.1 Use Cases Diagram ........................................................................................ 19
      6.4.2 Class Diagram .................................................................................................. 20
      6.4.3 Component Diagram ...................................................................................... 20
      6.4.4 Sequence Diagrams ...................................................................................... 21

7 Implementation and Testing ............................................................................................ 22
   7.1 Resource Allocation Algorithm .............................................................................. 22
   7.2 Simulator Features ................................................................................................. 24
      7.2.1 Scenario Configuration .................................................................................. 24
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.2.2</td>
<td>Scenario Simulation</td>
<td>24</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Exporting and Importing Scenarios</td>
<td>24</td>
</tr>
<tr>
<td>7.2.4</td>
<td>Simulation Reports</td>
<td>24</td>
</tr>
<tr>
<td>7.3</td>
<td>Scenario Configuration</td>
<td>25</td>
</tr>
<tr>
<td>7.4</td>
<td>Simulation Reports</td>
<td>27</td>
</tr>
<tr>
<td>7.5</td>
<td>Testing</td>
<td>29</td>
</tr>
<tr>
<td>8</td>
<td>Experimentation and Evaluation</td>
<td>30</td>
</tr>
<tr>
<td>8.1</td>
<td>Evaluation Approach</td>
<td>30</td>
</tr>
<tr>
<td>8.2</td>
<td>Predicted Outcomes</td>
<td>30</td>
</tr>
<tr>
<td>8.3</td>
<td>Simulation Results</td>
<td>30</td>
</tr>
<tr>
<td>8.3.1</td>
<td>Scenario 1: Posted Offer with Reputation Metrics</td>
<td>31</td>
</tr>
<tr>
<td>8.3.2</td>
<td>Scenario 2: Reverse Auction with Reputation Metrics</td>
<td>33</td>
</tr>
<tr>
<td>8.3.3</td>
<td>Comparison between Posted Offer and Reverse Auction Curves</td>
<td>36</td>
</tr>
<tr>
<td>8.4</td>
<td>Threats to Validity</td>
<td>36</td>
</tr>
<tr>
<td>9</td>
<td>Project Management</td>
<td>37</td>
</tr>
<tr>
<td>9.1</td>
<td>Risk Management</td>
<td>37</td>
</tr>
<tr>
<td>9.2</td>
<td>Development Process and Schedule</td>
<td>37</td>
</tr>
<tr>
<td>9.3</td>
<td>Backup Management</td>
<td>37</td>
</tr>
<tr>
<td>10</td>
<td>Conclusions and Future Work</td>
<td>38</td>
</tr>
<tr>
<td>Bibliography</td>
<td></td>
<td>39</td>
</tr>
<tr>
<td>Appendix</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>Appendix A: Test Case Specification</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>Appendix B: Other Simulator Screenshots</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>Appendix C: Instructions to Run the Tool</td>
<td></td>
<td>46</td>
</tr>
<tr>
<td>Appendix D: Example of XML Export File of an Scenario</td>
<td></td>
<td>47</td>
</tr>
</tbody>
</table>
List of Figures

Figure 1 High Level Description of the Implemented Model ............................................................... 2
Figure 2 Types of Delivered Services in the Cloud .................................................................................. 3
Figure 3 Supply and Demand Curves ........................................................................................................ 5
Figure 4 Architecture of a Reputation System ......................................................................................... 10
Figure 5 Some Core Classes in CloudSim ............................................................................................... 13
Figure 6 More Important Classes in CloudSim ......................................................................................... 14
Figure 7 Market-Based Cloud Resource Allocation Model .......................................................................... 16
Figure 8 Posted Offer Auction without Reputation Metrics ........................................................................ 17
Figure 9 Posted Offer Auction with Reputation Metrics ............................................................................. 18
Figure 10 Use Case Diagram of the Simulator ......................................................................................... 19
Figure 11 Class Diagram of the Simulator ............................................................................................... 20
Figure 12 Component Diagram of the Simulator ...................................................................................... 20
Figure 13 Sequence Diagram for Posted Offer Auction ........................................................................... 21
Figure 14 Sequence Diagram for Reverse Auction ................................................................................... 21
Figure 15 Screenshot of Seller Agents’ Configuration ............................................................................... 25
Figure 16 Screenshot of Trading Mechanisms’ Selection .......................................................................... 26
Figure 17 Screenshot of Scheduled Jobs’s Configuration ........................................................................... 27
Figure 18 Screenshot of Allocated Jobs per Seller Agent Report ............................................................... 27
Figure 19 Screenshot of Final Seller Agent’s Reputation Report ............................................................... 28
Figure 20 Screenshot of SLA Violations per Seller Agent Report ............................................................... 28
Figure 21 Screenshot of Simulation Log .................................................................................................... 29
Figure 22 Seller Agents for Evaluation ..................................................................................................... 30
Figure 23 Groups of Jobs for Evaluation ................................................................................................... 31
Figure 24 Curve of Posted Offer Auction with Reputation Metrics .......................................................... 33
Figure 25 Curve of Reverse Auction with Reputation Metrics ................................................................. 35
Figure 26 Curves of Posted Offer and Reverse Auction with Reputation Metrics ..................................... 36
Figure 27 Project Schedule ...................................................................................................................... 37
Figure 28 Screenshot of Importing / Exporting Menu ............................................................................... 44
Figure 29 Screenshot of Exporting a Scenario .......................................................................................... 44
Figure 30 Screenshot of Importing a Scenario .......................................................................................... 45
Figure 31 Screenshot of About Simulator Dialog ..................................................................................... 45
List of Tables

Table 1 Trading with Simultaneous Decisions ................................................................. 6
Table 2 Trading with Sequential Decisions ................................................................. 7
Table 3 Buyer and Seller Record Sheets in a Double Action ........................................... 10
Table 4 Posted Offer Simulation Results with Weight for Reputation of 0.99 ..................... 31
Table 5 Posted Offer Simulation Results with Weight for Reputation of 0.75 ..................... 32
Table 6 Posted Offer Simulation Results with Weight for Reputation of 0.5 ....................... 32
Table 7 Posted Offer Simulation Results with Weight for Reputation of 0.25 ..................... 32
Table 8 Posted Offer Simulation Results with Weight for Reputation of 0.0 ....................... 32
Table 9 Reverse Auction Simulation Results with Weight for Reputation of 0.99 .................. 33
Table 10 Reverse Auction Simulation Results with Weight for Reputation of 0.75 ............. 34
Table 11 Reverse Auction Simulation Results with Weight for Reputation of 0.5 ............... 34
Table 12 Reverse Auction Simulation Results with Weight for Reputation of 0.25 ............. 34
Table 13 Reverse Auction Simulation Results with Weight for Reputation of 0.0 ............... 35
Table 14 Risk Assessment ............................................................................................... 37
Table 15 Test Case Specification in Accordance with IEEE Standard 829-1998 ............ 40
Table 16 Test Case Specification in Accordance with IEEE Standard 829-1998 (Continue) 41
Table 17 Test Case Specification in Accordance with IEEE Standard 829-1998 (Continue) 42
Table 18 Test Case Specification in Accordance with IEEE Standard 829-1998 (Continue) 43
1 Introduction

Cloud Computing supplies IT services in a layered architecture where the bottom layer is referred to Infrastructure as a Service (IaaS); the middle layer to Platforms as a Service (PaaS) and the top layer to Software as a Service (SaaS). The relevance of the Cloud approach to provide these utility services is that customers pay to Cloud providers based on their usage of these utility services with the additional benefit of dynamically scaling up or down on demand. In a recent paper from the University of Melbourne Cloud Computing was called as the 5th Utility after water, electricity, gas and telephony [10].

Cloud users perform dynamic workloads in the Cloud and Cloud providers must be capable of fulfilling these expectations. Hence, it is important to define the expected QoS parameters in a SLA to guarantee an acceptable service performance for both parties. One idea to reduce the SLA violations in the Cloud is to make the analogy between a Cloud and an electronic marketplace [11]. Market-Based Systems are designed to improve the resource allocation and control in dynamic distributed systems by creating autonomous seller and buyer agents that carry out their Cloud jobs trading based on market models and financial strategies. In addition to this market oriented approach, we propose the use reputation metrics which are indicators to measure the dependability of a Cloud provider.

In order to experiment with a Market-Based Cloud we will analyze centralized and decentralized approaches to carry out the resource allocation. Additionally, we will provide an analysis of market mechanisms such as Posted Offers, Auctions and Bargains to be implemented as the autonomous trading mechanisms in the market negotiation. As reported by Experimental Economics [12], all these market experiments are focus on the predictions of neoclassical price theory, both Cloud providers and Cloud users need to estimate in advance the price for a Cloud job in the market. This price is the sum of the production cost plus the desired profit for performing the Cloud job. For this work, the estimation will be simplified by the assumption that the production cost of executing a job can be known in advance. However, the minimum and preferred profit of each Cloud provider is private and established according to his own preferences.

Carrying out all these experiments in real Cloud environments would be costly and time consuming. Moreover, the degree of complexity to reproduce the results under varying resources configuration and workloads would make the conclusions less confident. So, the best alternative to overcome these issues is to use a Cloud simulation framework. We have chosen the CloudSim [5] to extend it and develop the simulation tool to run our experiments. CloudSim is a generalized and extensible framework developed by the University of Melbourne that provides all the basic structure of a real Cloud architecture and performs the simulation using an event oriented approach [27].

Our simulation tool is fully documented using UML, and it is designed to be extensible to other market mechanism in order to allow new experiments in the future with new algorithms. In addition to Cloud users and Cloud providers, our model [3] introduces additional autonomous agents called Cloud brokers which are capable to perform the negotiation for the resource allocation for Cloud jobs on behalf of the Cloud users. A high level description of the Market-Based Cloud with reputation metrics can be described by the following graphic:
The main contributions of this work are:

1. Build a simulation tool by extending CloudSim to evaluate the resource allocation in Market-based Clouds
2. Provide the design and implementation of a trading algorithm that links the reputation metrics ideas with the Market-Based Systems approach to evaluate to what extent they improve the resource allocation in Cloud Computing environments.
3. Show a graphical view of the simulation results to analyze the impact of reputation metrics and a given market mechanism in Cloud architectures.

The rest of the document is organized as follows: first, a background section where a general description about Cloud Computing, CloudSim, Market-Based Cloud and reputation metrics is presented. Following that, comprehensive details related to the solution design to extend CloudSim API are provided. This section ends with a brief overview of the trading algorithms to implement the chosen market mechanisms. Then, the next section describes the scenarios to simulate, shows the experimentation results, its evaluation and a discussion of issues and limitations found in the simulations. Finally, the paper ends with a summary, conclusions and a suggestion for future works.
2 Background and Related Work

2.1 Cloud Computing

Cloud Computing is a reference model to provide IT infrastructure and applications as a service in a scalable manner. It is aimed at optimizing the resource utilization according to customized SLAs by means of virtualization and sharing resources. One formal definition was given in a report from the University of Melbourne as: “Cloud Computing is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified resources based on a service level agreement” [13].

Cloud computing comprises three IT layers: infrastructure, platforms and software. All of them are delivered as services over the Internet, hence providers are classified as:

- Infrastructure as a service (IaaS): offering web-access to processing, storage or connectivity. The key point is that hardware resources are abstracted and encapsulated by virtualization. However, end users have control over operating systems, storage and applications.
- Platform as a Service (PaaS): offering easy development environments, reusable components, libraries, collaboration services and workflow facilities to design, develop, test, deploy and host applications.
- Software as a Service (SaaS): offering services directly consumable by end-users. The main difference from conventional software suppliers is the deployment, licensing and billing model.

The Cloud involves three major actors: the Cloud user, the Cloud vendor and the original Cloud provider. The cloud vendor is an organization that has a local tax registration and provides the Cloud services to the Cloud user according to expected levels of quality of experience (QoE) and quality of service (QoS) based on a service level agreement (SLA). However, the Cloud vendor is not necessarily the owner of the infrastructure, platform or software offered to the Cloud user. Here appears the third actor, the original Cloud provider, who is the organization who owns the SaaS, PaaS or IaaS.

Clouds which are available to consumers in a pay-as-you-go manner are called Public Clouds or external clouds, while clouds which are owned, controlled and used by a single organization are denominated as Private Clouds or internal clouds. Single clouds from these types can be combined to get Hybrid Clouds and Federation of Clouds. Hybrid clouds are an alternative to get scalable IT resources provided by external clouds and to get security and privacy provided by internal clouds, at the same time. Federation Clouds are a group of single clouds collaborating to exchange data and processing resources through interfaces.

![Diagram of Cloud Computing Services](image)

Currently, there is a wide range of classification for Clouds. However, one of the most complete is established according to e-Science [1]. Here, Cloud taxonomy can be classified in the following categories:
1. **Business model**: this subtaxonomy classifies clouds into: Infrastructure as a Service, Platform as a Service, Software as a Service and Storage as a Service. The last one is a service that offers structured manners to access storage and databases remotely located. It is chiefly used in scientific experiments.

2. **Privacy**: this classifies clouds as: Public such as Google and Amazon, Private and Mixed such as Hybrid or Federated Clouds.

3. **Pricing**: based on this category, clouds are considered as: Pay-per-use and Free.

4. **Architecture**: According to architecture subtaxonomy, key cloud attributes classifies the clouds. These are: Heterogeneity, Virtualization, Security, Resource Sharing, Scalability and Monitoring.

5. **Technology infrastructure**: it takes into consideration the computational power of the cloud. Hence, there two categories: High-Performance computing (HPC) support and non HPC support.

6. **Access**: it establishes four kinds of access criterion: Mobile clients, Web browser, Thin clients and APIs

7. **Standards**: it classifies clouds based on the protocols and standards that are used for Security, Data, Virtualization, Syndication and Communication.

8. **Orientation**: based on this subtaxonomy, clouds are divided as Task-centric or User-centric.

Some of the Cloud Computing environments available in the IT industry are Amazon EC2, Microsoft Azure and Google App Engine. A comparison between six representative Cloud environments with industrial linkages is detailed in [10].

### 2.2 Market-Based Systems

People have been using markets to trade goods and services for thousands of years. A market is the natural environment where autonomous agents such as sellers and buyers carry out their trading without a global controller. In [14], market based systems are defined as “any system that uses the concept or certain features found in a market”.

A key characteristic of the Market-Based Systems is that they have no need of a central coordinator or global controller. In fact, agents in the system have no way to know all the parameters and states in the market. As a result, the interaction among individual agents leads to a global optimization by adopting an appropriate economic trading behavior.

Market economics are applied in Computer Science to find new methods to define decentralized allocation policies for scarce resources among multiple agents, known as Market-Based Resource Allocation (MBRA) and to apply mechanisms of control, known as Market-Based Control (MBC) [11]. A Market-Based System is composed by

1. **Traders**: the components responsible of quoting prices and performing the haggling process; and

2. **Marketplaces**: the places where buyers and sellers meet in order to reach an agreement.

Market-Based Systems implement buyer agents who need to perform a job and have money to pay for it. Also, implement seller agent who own resources and compete with others sellers to get the job execution on their resources and charge to the buyer for it. All this trading obeys the two basic rules: when the demand surpasses supply, price will rise (seller’s market). On the other hand, when supply surpasses demand, price will fall (buyer’s market). In general we are familiarized with the idea that the price is a function of the demand or supply [11]. This demand curve is essentially a graph of price vs. quantity to represent the price that buyers would for a quantity as it is shown in the figure. In the same way, the supply curve can be drawn on a graph of price vs. quantity to represent the price that sellers are ready to sell the goods or services as it is also shown in the picture.
2.3 Market-Based Cloud

A Market-Based Cloud is a Cloud environment implemented according to the definition for a Market-Based System. The main components in a Market-Based Cloud are:

1. **Cloud users** request job executions at an expected QoS parameters specified by a SLA,
2. **Cloud providers** or sellers own processing resources to trade,
3. **Cloud brokers** or buyer perform the Cloud trading on behalf of the Cloud users to select the most suitable Cloud provider for the job.
4. **Market mechanism**: establishes a set of rules to the trading interactions between buyers and sellers in order to define how buyers quote bid prices and seller quote offer prices.

This approach is aimed at analyzing to what extent the resource allocation based on market-based controls and reputation metrics can minimize SLA violation in Cloud architectures.

There are two approaches for managing resource allocation: centralized and decentralized [4]. In the first one, resource allocation is performed by a single node or a coordinator which is responsible for analyzing the user requirements and evaluating the current Cloud architecture configuration in order to decide how to optimize the utilization of resources from a global perspective. The key disadvantages for this classical approach include:

- ✔ The Cloud architecture must be static while the central coordinator is carrying out the calculus of the resource allocation algorithm
- ✔ The central coordinator knows the global state of the resources in the Cloud and
- ✔ The central coordinator must be connected to every single resource which leads to scalability problems.

On the other hand, a decentralized resource allocation mechanism is carried out by multiple self-interested agents. Each of them acts on behalf of a user at the decision making level while trying to maximize its own good without concern for the global benefit.

The main advantages of this mechanism are: no single point of failure, no performance bottleneck and the amount of exchange of messages to converge on an optimal solution.

Some distributed trading mechanisms are auctions and bargaining.
2.4 Experimental Economics

Economic theories are aimed at explaining market activity. Experimental economics [12] is the systematic evaluation of these economic theories in order to prove or disprove them by modeling experiments under controlled environments. Economics experiments were classified by Edward Chamberlain (1948) in three branches:

1. **Market experiments**: are aimed at analyzing the predictions of neoclassical price theories.
2. **Game experiments**: are focused on analyzing environments where each participant is playing a best response to the strategies choices of other participants.
3. **Individual decision making experiments**: are conducted on environments where uncertainty comes from external random events and not from other agents’ decisions.

This paper focus on market experiments were trading carried out by buyers and sellers is autonomous and decentralized.

The principal advantages of market experiments are the replication and control. Replication refers to the capacity to reproduce the experiment and get some tangible evidence to support the conclusions. Control refers to the capacity to configure the scenario in consonance with the required conditions to evaluate the theory. On the other hand, chief drawbacks of market experiments are sophistication and complexity. Sophistication because decision makers in the economy are more sophisticated than experimentation agents which may lead to some degree of uncertainty in the conclusion. The other drawback is complexity due to the degree of complexity achieved for an environment configuration may be simpler than real scenarios in the market.

Trading mechanisms can be classified in two categories of price competition: models with simultaneous decision and models sequential decisions. The following tables taken from [12] summarize the behavior of these trading mechanisms:

<table>
<thead>
<tr>
<th>Trading Mechanism</th>
<th>#Sellers/#Buyers (# units)</th>
<th>Who proposes prices</th>
<th>Decision and Timing</th>
<th>How contracts confirmed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cournot (quantity choice)</td>
<td>-/-</td>
<td>Price is endogenous</td>
<td>Quantities posted simultaneously</td>
<td>Simulated buyers</td>
</tr>
<tr>
<td>Posted Offer Auction</td>
<td>-/-</td>
<td>sellers</td>
<td>Offers posted simultaneously</td>
<td>Buyers shop in sequence</td>
</tr>
<tr>
<td>Ultimatum Bargaining (offer version)</td>
<td>1/1</td>
<td>sellers</td>
<td>Seller makes single offer on 1 unit</td>
<td>Buyers accepts or reject</td>
</tr>
<tr>
<td>Posted Bid Auction</td>
<td>-/-</td>
<td>buyers</td>
<td>Bid posted simultaneously</td>
<td>Sellers shop in sequence</td>
</tr>
<tr>
<td>Discriminative Auction</td>
<td>1/- (N units)</td>
<td>buyers</td>
<td>Bid posted simultaneously</td>
<td>Highest N bidders pay own bids</td>
</tr>
<tr>
<td>1st Price Sealed-Bid Auction</td>
<td>1/- (1 Unit)</td>
<td>buyers</td>
<td>Bids posted simultaneously</td>
<td>High bidder pays own 1st price</td>
</tr>
<tr>
<td>Competitive Sealed-Bid Auction</td>
<td>1/- (N units)</td>
<td>buyers</td>
<td>Bids posted simultaneously</td>
<td>Highest N bidders pay N+1st price</td>
</tr>
<tr>
<td>Second price Sealed-Bid Auction</td>
<td>1/- (1 unit)</td>
<td>buyers</td>
<td>Bids posted simultaneously</td>
<td>Highest bidder pays 2nd price</td>
</tr>
<tr>
<td>Clearinghouse Auction</td>
<td>-/-</td>
<td>Buyers and sellers</td>
<td>Bids and offers posted simultaneously</td>
<td>Intersection of bid and offer arrays</td>
</tr>
</tbody>
</table>

*Table 1 Trading with Simultaneous Decisions*
Table 2: Trading with Sequential Decisions

<table>
<thead>
<tr>
<th>Auction Type</th>
<th>#Sellers/#Buyers (# units)</th>
<th>Who proposes prices</th>
<th>Decision and Timing</th>
<th>How contracts confirmed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dutch auction</td>
<td>1/- (1 unit)</td>
<td>Seller clock</td>
<td>Price lowered sequentially</td>
<td>Buyer who stops clock</td>
</tr>
<tr>
<td>English auction</td>
<td>1/- (1 unit)</td>
<td>Auctioneer</td>
<td>Prices raised sequentially</td>
<td>Sale to high bidder</td>
</tr>
<tr>
<td>Bid auction</td>
<td>-/-</td>
<td>Buyer</td>
<td>Prices raised sequentially</td>
<td>Sellers</td>
</tr>
<tr>
<td>Offer auction</td>
<td>-/-</td>
<td>Sellers</td>
<td>Prices lowered sequentially</td>
<td>Buyers</td>
</tr>
<tr>
<td>Double auction</td>
<td>-/-</td>
<td>Both types</td>
<td>Bids raised and offers lowered sequentially</td>
<td>Both types</td>
</tr>
<tr>
<td>Decentralized auction</td>
<td>-/-</td>
<td>Both types</td>
<td>Sequentially but decentralized</td>
<td>Both types</td>
</tr>
</tbody>
</table>

2.4.1 Auctions

An auction is defined by Krishna in [15] as “one of the many ways that a seller can use to sell an object to potential buyers with unknown values”. In an auction the price of the object is the result of a competition among the buyers based on the rules previously defined by the seller. An auction has a wide range of variants, for example: the seller may post a fixed price and sell the object to the first interested buyer, or the seller may pick up a buyer randomly and negotiate the price with him, or the seller may set out an auction and then negotiate with the winner, etc. In all the cases, the seller always tries to maximize his own benefit without any concern about the social good. He wants to sell a product and get the highest possible payment for it while the buyers aim at getting the product at the minimum possible price. Depending on the mechanism by which the bidder establishes the value of a product, there are three kinds of auction settings:

1. **Private value auction**: the value of the product depends only on the agent’s own preference. One valid assumption is that the agent knows in advance the value of the product.
2. **Common value auctions**: the product value determined by an agent depends on the value established by other agents.
3. **Correlated value auctions**: it is a combination between the two previous settings. The value of a product depends partly on agent’s own preferences and partly on others’ values.

Auction setting produce changes in the properties of each auction protocol. Some of the auctions protocols are:

1. **English auction**: each bidder raise his bid and when anyone else is interested in make more bids or offers, the auction finish and the product is sold to the bidder that made the bid with the highest. This kind of auction is also called First-price open-cry.
2. **First-price sealed-bid auction**: each bidder makes his offer with no knowledge of the other’s bids. After all the offers are received, the product is sold to the bidder with the highest price in his bid.
3. **Dutch auction**: the auctioneer sets a price for the product higher than the current one. Then during the auction process, he is decreasing the price until one of the bidders makes a bid with the current price. This type of auction is also called descending auction.
4. **Vickrey auction**: each bidder makes his offer without any knowledge about the others’ bid. The bidder who made the offer with the highest price wins the auction but at the price of the second highest bid. It has two desirable results. First, the bidders make their truthful offers which leads to efficiency in the global process. Second, the bidders have no need to guess the value of the other bid because they have no relevant importance in making the bidding decision. Vickrey auction are widely used in computational multi-agent systems such as resource allocation in operating systems, bandwidth allocation in networks and to computationally control heating in buildings. However,
Vickrey auctions have not been widely adopted for human auctions. This auction type is also known as Second-price sealed bid.

5. **All-pay auction**: each bidder participating in the auction must pay an amount of his bid to the auctioneer. It is mainly used in resource reallocation by multi-agent systems.

### 2.4.2 Posted Offer Auction

Posted Offer Auction is a trading mechanism where sellers post their list of prices in a take-it-or-leave-it approach as it is done for example in retail markets. After a seller autonomously sets a price for a product, the potential buyers are expected to make an offer for it. A characteristic to be highlighted in this mechanism is that sellers have absolute influence over the market price because as the seller posts a take-it-or-leave-it price offer, the buyers are not able to negotiate the price. As a result, the market usually converges from prices higher than the equilibrium [8]. The opposite approach, called Posted Bid Auction, is gotten by inverting the roles so that sellers are invited to make desired sales at the bids posted by buyers.

The posted offer auction is based on a large number of repeated series of periods. For each period, the trading is composed by two phases. First, each seller autonomously announces a price for the period and the maximum number of units to be offered at that price. Each seller’s price (but not the maximum quantity) is shown to buyers and the other sellers, once these have also completed their own decision making processes for quoting prices. Then, when all sellers have quoted their prices, a buying sequence starts. Without any order in particular, buyers are sequentially given the opportunity to make their purchases from sellers at the specified prices. Buyers can make their trading with any seller as long as this one hasn’t reached his maximum sales quantity. This process finishes when all buyers have made their shops or all the available units have been sold.

As it is shown in experiments carried out in [12], both the stability and the structure of supply and demand curves in a posted offer auction usually generate a non-competitive outcome. What is more, experiments prove that posted offer prices converge slowly to conditions of excess supply. Finally, as buyers have a reduced capacity to lead the changes in reservation values (the highest value a buyer is willing to pay for an object), posted offer markets have a poor reaction to demand-side adjustments. On the other hand, these markets are very responsive to supply-side shocks.

### 2.4.3 Reverse Auction

Reverse auction is described in [17] as an auction where “suppliers are bidding to supply an object or service to a buyer and are pushing the prices down. [It is] likely to be most used where standardization of products/services is desired, markets are fragmented, transaction costs are high and global sourcing expertise is required”. So, a reverse auction is not recommended for highly customized specifications where just a few numbers of sellers have the capacity to offer the object or service. A good checklist for the conditions to apply reverse auction as the market mechanism is given in [16] and they can be summarized as:

1. **Spend characteristic**: generic
2. **Specifications**: defined by industry standard
3. **Market driver**: Mostly price/volume
4. **Industry competitiveness**: very competitive (commoditized)
5. **Supplier availability**: very many (large pool)
6. **Riskiness of switching suppliers**: low
7. **Supplier relationship**: transactional
8. **Switching cost**: negligible
9. **Level of detail for supplier qualification**: low
10. **Supply lead time**: short/immediate
11. **Spend examples**: no custom packaging; bulk chemicals; lab supplies

Reverse auctions are a mechanism that leads to a healthy competition among participants by giving them a clear insight into true market conditions. Their major benefits are:
1. **Time**: the negotiation process takes less time, which increases the productivity.
2. **Money**: it saves costs by increasing the negotiation rate with big amounts of sellers.
3. **Trust**: all the information is equally available for the participants at the same time which leads to a competitive process under openness and fairness conditions. This transparency gives a clear view of the trading process.

### 2.4.4 Bargaining

Bargaining is defined in [18] as “a situation in which individuals have the possibility of concluding a mutually beneficial agreement, there is a conflict of interests about which agreement to conclude and no agreement may be imposed on any individual without his approval”. In a bargaining a point of interest is the relationship between its outcome and the characteristics of the scenario.

In a bargaining mechanism, agents can make a mutual beneficial agreement, but they may have a conflict of interest about the terms of the agreement to reach. Bargaining in real world is composed by a finite number of competing agents so it is the middle between a monopoly and a perfect competition. There are two approaches for bargaining:

1. **The Axiomatic approach**: This approach is the most commonly used and it is based on the Nash solution, proposed by Nash (1950). It describes a set of N bargainers. They either achieve an agreement or fail to achieve it which leads a disagreement event. An agreement may be a price or any other contract that defines the action to be performed by the parties in each of the contingencies. The framework describes a bargain as a place where some axioms or desirable characteristics of a solution are postulated and then the solution concept that matches with the axioms is accepted. On the basis of Nash solution, the axioms are: Pareto optimality, symmetry, independence of irrelevant alternatives, and invariance to linear transformations of utility [12].

2. **The Strategic approach**: it is an approach that models the bargaining mechanism as an uncooperative game where the solution is given by the model’s equilibrium.

### 2.4.5 Double Auction

Double auctions so-called open outcry markets are described in [8] as: “Buyers and sellers post prices simultaneously, bids and asks, respectively, and whenever there is match, trade occurs. At any point in the trading period, buyers and sellers can observe the highest bid and lowest ask (or all bids and asks), accept an outstanding bid/ask or post new offers, which are queued and could be accepted in the future”.

On one side, in auctions with multiple buyers and a single seller, the buyers raise price bids until one interested remain which leads to an ascending-bid auction. On the other side, in auctions that a single buyer receives offers from competing sellers trying to sell a fixed number of objects, prices offers tends to being reduced until there is no excess of supply. On double auctions, these two opposite processes take place at the same time and trades occur during this negotiation.

Double auction creates a competitive market in prices and quantities. In fact, it appears to create these competitive results more quickly and reliably than any other market mechanism. Hence, when it comes to evaluation of any other trading mechanism, the comparison is usually performed against a double action as a standard mechanism.

Double action is composed by a sequence of trading intervals or periods, where each of them last a pre established duration depending on the volume of participants in the market. At the beginning of the trading interval, buyers receive unit valuations and sellers receive unit costs. After this, buyers are usually required to purchase first higher-valued units and then lower-valued units. The profit of the buyers is calculated as the difference between the unit value and the purchase price. There are no earned profits on units not bought. On the other hand, sellers earn profit as the difference between the contract price and the unit costs. As production is usually to fulfill orders, costs due to unsold units are not taken into consideration. An example of a double action scenario and the following table showing the negotiation carried out in one of the periods is given in [12]:
For experimentation purposes, only profitable actions are permitted in order to avoid bids above unit values or offers below unit costs. Otherwise, trades at a loss produce noisy signals to the market like predatory pricing, for example, which may be caused by the effort from one competitor to take another away from the business.

Markets organized under double auction mechanism are robust to a wide variety of supply and demand configurations, to limitations on the number of agents and to conditions determining communications between sellers.

### 2.5 Reputation Systems

Reputation is a collection of processed information about the performance of one agent as it was experienced by other who had a transaction with him. Reputation systems are described in [19] as “an important block for achieving trust within large distributed communities, especially when mutually unknown agents engage in ad-hoc transactions. Based on the knowledge of the past behavior it is up to each agent to form his opinion on his potential transaction partner”. An important mentioned term is trust, which is the degree of willingness to carry on an action that put agents on risk of harm and is based on an evaluation of the risks, rewards and reputation related to all the agents involved in the situation.

The rating in a reputation system is a client agent’s opinion of a server agent after his request was completed by the server agent. The architecture of a reputation system is represented by the following figure according to [19]:

![Figure 4 Architecture of a Reputation System](image-url)
The architecture is composed by the following actors and components: The ratee or agent to be rated. The collector receives ratings from agents known as raters. This information is processed and amassed by the processor. The algorithm used by the processor to calculate an accumulated representation of an agent’s reputation is the metric on the reputation system. The emitter publishes the results to other requesting agents.

It is worthwhile to highlight that a reputation is context dependant, which means, for example, that a reputation as IaaS provider cannot be applied in another context such as SaaS provider. On each independent context, a reputation system model is based on its specific metrics to compare different agents by measuring their reputation values. This measuring used for these metrics in reputation systems, classify the models in:

1. **Accumulative Systems**: system collects and amasses all the ratings in order to calculate the overall reputation of an agent. It means that metrics reflects the trustworthiness of an agent based on the number of good behaviors.
2. **Average Systems**: systems calculate the agent’s reputation as the average of all rating the agent has got. The idea behind of this metric is that an agent is going to behave in the same way most of the time.
3. **Blurred Systems**: reputation systems compute a weighted mean sum of all ratings. Newer ratings have more influence for the current reputation. This metric is based on the fact that agents may change their behavior during their lifetime.
4. **OnlyLast Systems**: this kind of systems computes the last gathered rating for an agent. In fact, it is an extreme case of the Blurred system.
5. **EigenTrust Systems**: systems combine the agent’s local reputation values iteratively to a global reputation. Local reputation is defined by a set of agents such that an agent may have a different reputation on each local reputation system. Whereas in a global reputation system there is a single value for each agent.
6. **Adaptive Systems**: systems behave differently to the received rating depending on the current reputation value of the agent. For instance, if the current reputation of the agent is low then a positive rating causes a big increment in the reputation. On the other hand, if the current reputation is high then the same positive rating produces only a small increment in the reputation. It may behave similarly for negative ratings.
7. **Beta Systems**: systems try to predict the future behavior of an agent by means of statistics. This prediction takes into consideration the past behavior of the agent and the probability to behave good or bad in the next request.

### 2.6 SLA in Cloud Architectures

The Cloud is a source of services on the basis of contracts that assure satisfaction of the Quality of Service (QoS) parameters of Cloud users conforming to specific service level agreements (SLAs). According to [20] the use of a SLA is intended to “define a formal basis for performance and availability the provider guarantees to deliver. SLA contracts record the level of service, specified by several attributes such as availability, serviceability, performance, operation, billing or even penalties in the case of violation of the SLA. Also, a number of performance-related metrics are frequently used by Internet Service Providers (ISPs), such as service response time, data transfer rate, round-trip time, packet loss ratio and delay variance”.

Additionally, Cloud users and Cloud providers may determine cost and penalties on the basis of the achieved performance level in the so-called utility-based SLA. The Cloud resource allocation management may be included as revenues or penalties when QoS parameters are satisfied or violated.

Utilization of SLA in the Cloud may be intended to:

1. **Control objectives**: this use is aimed at regulating QoS metrics of interest such as CPU utilization, response time; or at optimizing QoS metrics to certain values specified by the Cloud user or Cloud
provider; or at rejecting unwanted workloads that might produce an unpredictable performance. It may also be a mix of the described objectives.

2. **Metrics and Adaptation Mechanisms**: this use is designed for controlling systems by using three kinds of metrics: system-level metrics such as CPU and memory utilization; application-level metrics such as response time and throughput; or business-level metrics such as profit in SLAs. However, adaptation mechanisms are required in the case of conflicting goals. The five principal mechanisms are: admission control to avoid excessive workloads, resource reallocations to manage dynamic reallocations; real-time scheduling of jobs; load balancing and content adaptation.

### 2.7 CloudSim

A challenge in Cloud Computing is the use of real Cloud environments such as Microsoft Azure, Google App Engine and Amazon EC2 for evaluating the performance of resource scheduling and allocation policy under varying resource configurations and user requirements. The results of the experiments on these dynamic environments are difficult to reproduce due to the cost and the unpredictable conditions in an Internet-based environment. For this reason, a valid alternative to overcome this challenge is to run the experiments on simulation tools.

CloudSim is described in [5] as: “a new generalized and extensible simulation framework that enables seamless modeling, simulation, and experimentation of emerging Cloud computing infrastructures and management services”. Some of its features are:

- A virtualization engine to manage independent virtualized services on a data center node,
- Allocation of resources based on space-shared and time-shared mechanisms. These features are the basis for extending the framework and developing more complex algorithms for Cloud Computing.

A further reason to use CloudSim is that existing Grid simulators such as GridSim, SimGrid and GangSim offer a platform to model and simulate Grid applications in distributed environments, but none of them provides support for the multi-layering service abstraction differentiation required by the Cloud (i.e. SaaS, PaaS, IaaS). Additionally, CloudSim tackles economic-driven resource management and application scheduling, which are the basis for developing more complex algorithms for economic entities such as Cloud brokers and Cloud exchange for trading of services in a market-based Cloud. However, the main problem of the framework is the lack of documentation and examples which requires a considerable effort to understand it by debugging and reviewing the source code.

#### 2.7.1 Simulation Approach

CloudSim API is an event and entity oriented simulation package. Its design decomposes the system under simulation in entities exchanging responsibilities each other by means of events. This way, the simulation clock progress is associated to the execution of these events.

#### 2.7.2 CloudSim Architecture

Event and entity driven simulation is supported in CloudSim by dozens of classes. However, only the most important are shown in the following diagram:
**SimEntity** is the abstract class that is used to represent entities in the framework. It is able to send and listen for an incoming event. The methods to be overwritten are startEntity, processEvent and shutdownEntity. This way, entities customize their behavior at initializing, destroying and performing the activities corresponding to the received event.

**SimEvent** is the class to represent the event that is exchanged between entities. It records the type of event, the generation time, scheduled time, finish time, source and destination and some commands to be executed at the destination entity or entities.

**FutureQueue** is the queue to hold all the events that have been generated by entities but with a delivery time that has not been reached yet by the simulation clock.

**DeferredQueue** is the queue holding all the events generated by entities and that have not been dispatched by its destination entity because it was busy at the arrival time.

**CloudSim** is the main class of the framework. It manages the event queues and controls the simulation clock to execute the events. On each step, scheduled events in the FutureQueue are removed and transferred to DeferredQueue. Then, events are executed by the responsible entity and removed from the queue.

The framework also offers support to model the objects in the Cloud such as host, data center, broker, virtual machine and so on. Additionally, it models management of memory, bandwidth and storage.

**Datacenter** models the infrastructure provided by Cloud providers. This way, it encapsulates host, memory, storage and so on. Datacenter manages the allocation of host to virtual machines by means of the defined policies in the class VmAllocationPolicy.

**Host** models a server in the data center and encapsulates all the related information such as CPU cores, memory and storage capacity. Also, it contains the policy to share the processing power and bandwidth among the virtual machines running on that host. For example, a core might be assigned per VM (space-shared policy) or the processing capacity of each core might be distributed among all the running VM (time-space shared), etc.

**Vm** is the class to represent a virtual machine being managed by a host in the Cloud. It encapsulates all the related information such as processor, memory, storage capacity and so on. An instance of CloudletScheduler is responsible for the policy of scheduling performed by a virtual machine to execute jobs or cloudlets.

**Cloudlet** is the class responsible for modeling a job or an application to be executed in the Cloud. A job is expressed by its computational requirements; this is a number of instructions. Also, it defines the length of data to be transferred through the network in order to estimate the bandwidth consumption.
DatacenterBroker models a Cloud broker that mediates negotiations between the Cloud user and the Cloud provider to allocate resources driven by the job’s QoS needs defined in a SLA. It represents the Cloud user taken the specific need of the users.

2.8 Related Work
There are some papers that have been considered to design the simulator proposed in this research. For instance, the article in [21] highlights the chief challenges inherent to the resource allocation in the Cloud and describes the resource allocation inputs: Cloud resources, resource modeling, application requirements and provider requirements. Another paper is [22], which proposes a real-time model consisting of a dynamic auction mechanism to improve the allocation problem in the Cloud. Finally, a research [3] still in progress is designing and exploring a market-based model that incorporates reputation metrics for managing SLA violations in Cloud architectures. So, our model takes and adapts these ideas, and based on them designs and implements a new model to improve the resource allocation in the environment of Cloud Computing.

3 Motivation
SLA violations in the environment of Cloud Computing are potentially caused by the gap between the Cloud user needs and the real capacity of a Cloud topology. In order to reduce this sort of violations, our core idea is to propose a new mechanism for dynamic resource allocation in the Cloud by combining concepts coming from market based systems and reputation systems. Market based systems apply pricing mechanism and autonomous trading among independent agents to achieve a global optimization. Additionally, reputation systems keep a record about the performance of specific targets at carrying out previous tasks to make predictions about a future behavior. In our experiments we will combine these two systems to design a novel model that matches more efficiently Cloud user’s requirements and Cloud providers’ offers based on price and reputation metrics. As, these experiments in a real environment would be expensive and would take a long time we have decided to implement a simulation tool to claim the results of our model.
4 Research Objectives
The objectives considered at the beginning of this work are listed below:

1. Design a new model to improve the Cloud resource allocation performance by means of market mechanisms and reputation metrics.
2. Develop a simulation tool that allows both GUI and file-based input parameters specification to experiment resource allocation mechanisms in the Cloud.
3. Implement an algorithm on the basis of the designed model and compare the predicted outcome with the simulation results.
4. Display graphically the simulation results in order to facilitate their analysis.

5 Challenges
The challenges of the current work may be summarized as is shown in the following list:

1. Learn about different market-based mechanisms and their use for resource allocation in Cloud environments.
2. Choose and implement an appropriate reputation system and its corresponding reputation metrics.
3. Understand a Cloud Computing simulation framework and then build on top of this a market-based Cloud simulator.
4. Understand experimental economics to design controlled environments and suitable market-based scenarios to evaluate how the different trading mechanisms may improve the resource allocation in the Cloud.

6 System Design
6.1 Model Description
The model focuses on analyzing the resource allocation performance in a Market-Based Cloud by using different trading mechanisms with and without reputation metrics. For this reason, an artificial semi-decentralized market with the following actors and components has been considered:

1. Cloud user: he submits a job to the Cloud with specific resource requirements, the maximum price he is willing to pay and the expected QoS parameters in a SLA.
2. Job: it is a task to be submitted in the Cloud and it is defined by a number of millions of instructions required to complete the task.
3. SLA: it contains a formal specification of the required QoS parameters such as priority, reliability, availability and so on.
4. Buyer agent: the trading to allocate a Cloud user’s job in the Cloud is carried out by a Buyer agent who acts on behalf of the user and his interest is to earn money per allocated job successfully executed.
5. Seller agent: he is the owner of a Cloud resource and he carries out the trading to allocate resources on his infrastructure.
6. Cloud resource: a resource in the model is thought as a datacenter with virtual machines on physical host with specific computing capacity defined by millions of instructions per second to provide the three service layers.
7. Region: the Cloud is composed by several regions so that each region behaves as an independent Cloud.
8. Trading mechanism: it is the set of rules that defines the market mechanism by which buyer and seller agents reach an agreement to allocate a job in the Cloud.
9. Reputation mechanism: it collects the ratings given to a seller agent per allocated job and it also evaluates the current seller’s agent reputation.
A graphical description of the model including its actors and components is shown in the following figure (Some icons taken from http://findicons.com):

In a typical scenario, our model consists of a set of regions $R$, an autonomous distributed trading mechanism $T$ and a reputation mechanism $R$. A region contains a set of Seller Agents $S$. Each seller agent owns resources that are available to provide layered services in the Cloud. We assume that the members of $S$ are self-interested. Additionally, the region has a large population of users $U$ and each user requires allocating a job in the Cloud to be executed as it is specified in a SLA. Finally, each region contains a pool of Buyer Agents $B$ to take the jobs and the SLAs submitted by users to the Cloud, these buyer agents follow the set of rules defined in $T$ and the reputation metrics provided by $R$ to negotiate and allocate the job on one of the Seller Agent’s resources.

A seller agent sets a price for the use of their resources. This price is expressed in terms of artificial money because as long as all the trading mechanisms are done in this money, the value of the money is not relevant. Seller agents always attempt to maximize their own benefit without taking the global good into consideration. They establish a price according to their own preferences but always over the production cost, which is the value of the effort to execute the job expressed in terms of money. The price upper limit is defined by the preferred profit and the lower limit is given by the minimum profit that the seller agent is willing to keep his interest into performing the job. A seller agent’s resources have a probability $p$ to succeed and a probability $1-p$ to succeed at executing a user’s job on his infrastructure.
Trading mechanisms are usually only centered on the product price. So, the model adds a reputation system to be considered in the negotiation process. The idea behind is that a seller agent with a high reputation is more likely to provide a high dependability on the offered service.

Using this trading mechanism, for each user’s job submitted to the system, a buyer agent performs the negotiation and chooses the best option to allocate the job. It trades with all the seller agents in his region and ponders the price of each offer with the corresponding reputation in order to carry out the calculation to get the best option. If the job execution fails the seller agent is given a bad rating. Then, the buyer agent performs a new resource allocation to a different seller agent in the same region. In case, there is no seller agent to execute the job, the buyer agent starts the negotiation to a different region.

After the job is successfully executed, the seller agent releases his resources, gets the payment for the performed job and is given an award to improve his reputation.

6.2 Model Assumptions
In order to focus more precisely on the core of the resource allocation problem, a number of simplifying assumptions that keep unaltered the underlying model behavior have been done:

1. That there is no collusion among seller agents,
2. That each seller agent sets up a price over the production cost,
3. That each buyer agent sets up a valuation not under the production cost,
4. That there is enough resources in the Cloud to satisfy any user’s job,
5. That each buyer agent is trading with seller agents to allocate only one job at a time,
6. That a buyer agent can negotiate with all the seller agents located in the same region, and
7. That network connectivity is uniform.

6.3 Trading with Reputation Metrics
The following graphic illustrates a typical posted offer auction without reputation metrics where a buyer submit a job to be done, then each seller post a price and finally the buyer chooses the best alternative only based on the lowest price. It must be highlighted that the price is given by the production cost plus a desired profit.

![Figure 8 Posted Offer Auction without Reputation Metrics](image)
On the other hand, in our posted offer with reputation metrics, the buyer agent uses an adjusted price instead of a normal price. The adjusted price is defined by the production cost plus the desired profit plus a penalty which is inverse proportional to the seller agent’s reputation.

Reverse auction with reputation metrics behaves similarly to posted offer with reputation metrics, but it has several rounds to try to push the prices down. So, the seller agents try to compensate a low reputation by reducing their expected profit in order to achieve a job on its infrastructure.
6.4 Simulation Tool Design

6.4.1 Use Cases Diagram
Use cases are their relationship is shown in the following figure:

![Use Case Diagram of the Simulator](image)

*Figure 10 Use Case Diagram of the Simulator*
6.4.2 Class Diagram
The following figure represents the class diagram that summarizes the main classes and their relationship utilized in the market-based model implementation.

![Class Diagram](image1)

**Figure 11** Class Diagram of the Simulator

6.4.3 Component Diagram
The following figure represents the component diagram that summarizes the main components used in the market-based model implementation.

![Component Diagram](image2)

**Figure 12** Component Diagram of the Simulator
6.4.4 Sequence Diagrams
The sequence diagrams for both market mechanisms implemented in the simulator are shown in the figures below:

Figure 13 Sequence Diagram for Posted Offer Auction

Figure 14 Sequence Diagram for Reverse Auction
7 Implementation and Testing

7.1 Resource Allocation Algorithm

Declarations

Given a collection of a \( p \) number of jobs, where each Job:

- Requires a number of CPU instructions to being executed in the Cloud. It is measured in MI (millions of instructions)
- Belongs to a Cloud user which specifies the maximum profit that he is willing to pay
- A memory space to be allocated (in GB)
- It has a SLA attached to it where they are specified the following parameters:
  1. Priority: High, Medium, Low
  2. Reliability: High (1), Medium (0.75), Low (0.25)
  3. The region where the Cloud user is located

Given a collection of \( n \) Seller Agents, where each one owns resources with:

- A CPU capacity in MIPS (millions of instruction per second)
- A memory space (in GB)
- A probability \( p \) to succeed and a probability \( 1-p \) to fail an agreed QoS parameter (In our model, this is Availability)

A Seller Agent has a reputation that might go from a minimum of 1 to a maximum of 50. These boundaries are aimed at guaranteeing a tradeoff between price and reputation. Otherwise, the reputation could become extremely high which would lead to covert the model in a simple reputation system. In the same way, if it is extremely low, the model would be a simple market-based system. The difference between the minimum and maximum boundaries is associated to the amount of the increment or decrement in the reputation given for a rating after completing with success or failure a job.

A Seller Agent makes a bid to perform a Job by means of:

- Price (Job) = Production Cost + Desired Profit

Given a pool of Buyer Agents with a preferred size \( m \), where a Buyer Agent calculates an Adjusted Price to a Price offered by a Seller Agent according to:

- Adjusted Price (Job) = Price + Penalty (Seller Agent’s Reputation)
- Penalty(Seller Agent’s Reputation) = Price \times \text{Increasing Price Factor}
- \text{Increasing Price Factor} = \text{Weight for Seller Agent’s Reputation} \times \text{Job Reliability in SLA} \times \text{Seller Agent’s Failed Jobs Rate} \times \left( \frac{\text{MAXIMUM_REPUTATION} - \text{Seller Agent’s current reputation}}{\text{MAXIMUM_REPUTATION}} \right)

The Weight for Seller Agent’s Reputation defines an upper boundary of the percentage of the penalty. It goes from 0 to 0.99 because it is a percentage aimed at finding a tradeoff between price and reputation. A weight of zero (0) means that the Adjusted Price is always going to be equal to the Offered Price. On the other hand, if the value is 0.99, it means that the penalty caused by the reputation is almost as important as the price.

Given a Market that contains a pool of Buyer Agents and receive each Job to assign it to an existing or new Buyer Agent
Given a Trading Mechanism that defines a set of rules for the price negotiation in the market. The mechanism might be Posted Offer or Reverse Auction

Given an Adaptive Reputation System which increases the reputation of a Seller Agent by 1 after completing successfully a Job if his current reputation is below the maximum allowed or decreases his reputation by 1 after failing at performing a Job if his current reputation is above the minimum allowed.

**Definitions**

Randomly initialize Resources and Seller Agents with a medium level reputation.

Randomly initialize each Job in the collection and dispatch them to the market

Define the Trading Mechanism for the market: Posted Offer or Reverse Auction.

**Begin**

Seller Agents are listening for any incoming Job in Distributed Trading Mechanism

Market receives incoming Jobs and for each of them assigns a free Buyer Agent in the corresponding region. If any Buyer Agent is free then creates a new one.

Buyer Agent submits the Job to the Distributed Trading Mechanism

Each Seller Agent performs a valuation of the Job and makes its offer to the Buyer Agent. This offer is the production cost plus the desired profit which oscillates from the minimum to the preferred profit.

Buyer Agent receives offers from interested Seller Agent for a period of time T. If there is no offer in the same Region, the Buyers start trading with Seller in the next Region.

Buyer Agent verifies its reputation in the central repository and calculates the Adjusted Price.

Buyer Agent picks the best offer depending on the distributed Trading Mechanism.

- In the case of Posted Offer, the lowest adjusted price is the best offer.
- In the case of reverse Auction, it starts a new negotiation round based on the lowest Adjusted Price in the previous round.

The resource allocation is performed according to the Buyer Agent selection.

The Seller Agent tries to perform the Job.

In case that the Job is successfully completed in accordance with the QoS parameters in the SLA (for example availability)

Increase Seller Agent’s reputation to the next level in case he hasn’t reached the maximum yet.

In case that the Job is successfully completed but without accomplishing the QoS parameters in the SLA

Decrease Seller Agent’s reputation to the next level in case he hasn’t reached the minimum yet.

In case that the Job is not completed conforming QoS parameters in the SLA

Decrease Seller Agent’s reputation to the next level in case he hasn’t reached the minimum yet. Then, reintroduce the task into the distributed Trading Mechanism to perform a new negotiation with other Sellers in the Region
The new reputation of the Seller Agent is recorded for new incoming jobs.

End

7.2 Simulator Features

The simulation tool has four main functionalities: configure a scenario through GUI; then simulate it and continue it from last simulation run; prepare and display the results in pie charts, bar charts and a log; and export/import a scenario for reproducing results. We can summarize the features in the following way:

7.2.1 Scenario Configuration

1. Add a default Seller Agent.
3. Delete a Seller Agent.
4. Choose the trading mechanisms to run the simulation and set the weight for reputation.
5. Add a default Group of Jobs.
6. Modify the attributes of a Group of Jobs.
7. Delete a Group of Jobs.

7.2.2 Scenario Simulation

8. Execute the configured scenario.
9. Close previous simulation results.
10. Continue the simulation by resending the jobs and keeping the Seller Agents’ reputation values from the last simulation.

7.2.3 Exporting and Importing Scenarios

11. Export seller agents, trading mechanisms and buyer agents to a XML file (An example of the exported file is in the appendix D).
12. Import seller agents, trading mechanisms and buyer agents from a XML file.

7.2.4 Simulation Reports

13. Allocated Jobs per Seller Agent - Pie Chart (Screenshot of these reports are in the appendix B)
14. Final Seller Agents’ Reputation - Bar Chart
15. SLA Violations per Seller Agent - Bar Chart
16. Log view to detail the simulation steps.
7.3 Scenario Configuration

A scenario is defined by a number of available Seller Agents, a group of scheduled Jobs to be sent to the Cloud and finally the trading mechanism to use to negotiate the resource allocation.

Each Seller Agent is implemented as a data center with one host which contains four cores. It is specified by the following attributes:

1. **Name**: the identifier of the Seller
2. **Reputation**: the initial reputation of the Seller. At the beginning all the Seller Agents must have the same reputation. (Reputation is limited from 1 to 50 to avoid distortions. For example, if a Seller Agent gets a high reputation the market would become a monopoly because it would be only a reputation system and the trading by price would make no sense).
3. **Minimum Profit**: the minimum profit at which the Seller Agent is willing to execute a Job on his infrastructure. It is a percentage of the production cost. So, it goes from 0.0 to 1.0. The lower limit ensures that the offered price will never be under the production cost.
4. **Preferred Profit**: the preferred profit that the Seller Agent would like to earn for performing a Job on his infrastructure. It is a percentage of the production cost. So, it goes from 0.0 to 1.0. The lower limit ensures that the offered price will never be under the production cost.
5. **Millions of Instructions per Core**: the processing capacity of each core in the Seller Agent infrastructure. It is of the expressed in millions of instructions per second.
6. **Probability to Succeed**: the probability \( p \) to perform a Job conforming to the specified SLA. As a consequence the probability to fail is \( 1-p \). The probability can go from 0.0 to 1.0.
7. **Region**: The region to which the Seller Agent belongs. There are four regions: North, South, East and West.

Then, the Posted Offer Auction, Reverse Auction or both can be chosen as the trading mechanisms used to negotiate the allocation of Jobs in one of the Seller Agent’s infrastructure. A trading mechanism is specified by:
1. **A weight for reputation**: the maximum percentage of the penalty that might be imposed to the price offered by a Seller Agent depending on his reputation in order to calculate the adjusted price. (The weight goes from 0 to 0.99 because it is a percentage of the price. For example, if the price is 100 and the weight is 0 the adjusted price would be equal to the original price because the reputation won’t be taken into consideration. On the other hand, if the weight is 0.99 the adjusted price might be up to 199 depending on the other parameters like Seller Agent reputation, reliability of the Job, etc.)

![Figure 16 Screenshot of Trading Mechanisms' Selection](image)

Finally, we define several Groups of Jobs. Each job in each group is assumed to belong to a different Cloud user. However, all the jobs in the same group share the same attributes. A Group of Jobs is specified by the following attributes:

2. **Group Id**: the Identifier of the group.
3. **Number of users**: the number of jobs generated in this group that will share the given characteristics for the group.
4. **Scheduled Time**: the time at which all the jobs in the group will be sent to the Cloud.
5. **Mi per Job**: the number of millions of instruction that each Job requires to be performed in the Cloud
6. **Mips per VM**: the processing capacity of the VM to be allocated in the Seller Agent’s infrastructure in millions of instructions per second.
7. **Minimum Reputation**: the minimum reputation required from a Seller Agent to be allowed to participate in the trading. It goes from 1 to 50.
8. **Region**: The region of the User who submits the Job to the Cloud. There are four regions: North, South, East and West.
9. **SLA Priority**: the priority for this Job specified in the SLA. The priority can be High, Medium or Low.
10. **SLA Reliability**: the reliability required for this Job and specified in the SLA. The reliability can be High, Medium or Low.
7.4 Simulation Reports
The simulation results are presented graphically and in a log that details the trading and resource allocation process as it is shown in the following figures:
Simulation Tool for Market-Based Cloud Resource Allocation

Figure 19 Screenshot of Final Seller Agent’s Reputation Report

Figure 20 Screenshot of SLA Violations per Seller Agent Report
7.5 Testing
Test case specification in accordance with IEEE Standard 829-1998 is shown in the appendix A. Additionally, unit testing has been carried out for important methods. These are included in the source code.
8  Experimentation and Evaluation

8.1  Evaluation Approach
We have used a controlled environment provided by a scenario running on the simulation tool in order to experiment with our model and get convincing evidence to prove that it can improve the resource allocation in the Cloud. Then, we will compare our predicted outcome against the simulation results.

8.2  Predicted Outcomes
We expect to find an inverse proportionality between the weight for reputation metrics in a trading mechanism and the number of SLA violation in a market based Cloud in order to claim that the use of reputations improve the resource allocation. On the other hand, if a clear pattern does not start to emerge, we will disprove our theory.

8.3  Simulation Results
We will detail and show the results from one of the scenarios that we have experimented. The scenario consists of 3 Seller Agents and 5 Groups of Jobs as it is shown in the figures below. In order to get a trend for the number of SLA violations with different Weights for reputation, we have simulated both trading mechanisms with the same scenario using 5 different weights for reputation: 0.0, 0.25, 0.50, 0.75 and 0.99.

![Seller Agents for Evaluation](image-url)
8.3.1 Scenario 1: Posted Offer with Reputation Metrics

In the case of Posted Offer, for every Weight for reputation we have run 6 simulations with 11 rounds each one, where each round takes the final Seller Agents’ reputation in the last round and resubmit the Jobs to the Cloud (Continue Simulation). This process produced a total of 55 jobs sent to the Cloud per simulation (5*11). Then, we got the total of SLA violations at the end of the last round in each simulation. Finally, we calculate the average of violations on each Weight for reputation in order to draw the curve SLA violation vs. Weight for reputation. The tables that summarize the simulation results for the differentWeights for reputation are presented:

Table 4 Posted Offer Simulation Results with Weight for Reputation of 0.99

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Jobs</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>40</td>
<td>45</td>
<td>50</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>Simulation 1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Simulation 6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.50</td>
</tr>
</tbody>
</table>

Figure 23 Groups of Jobs for Evaluation
## Simulation Tool for Market-Based Cloud Resource Allocation

### Table 5: Posted Offer Simulation Results with Weight for Reputation of 0.75

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Jobs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.50</td>
</tr>
<tr>
<td>Simulation 1</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>Simulation 2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Simulation 3</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 6</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.50</td>
</tr>
</tbody>
</table>

### Table 6: Posted Offer Simulation Results with Weight for Reputation of 0.5

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Jobs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9.17</td>
</tr>
<tr>
<td>Simulation 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>Simulation 2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>Simulation 3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Simulation 6</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9.17</td>
</tr>
</tbody>
</table>

### Table 7: Posted Offer Simulation Results with Weight for Reputation of 0.25

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Jobs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11.33</td>
</tr>
<tr>
<td>Simulation 1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>Simulation 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>Simulation 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>Simulation 6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11.33</td>
</tr>
</tbody>
</table>

### Table 8: Posted Offer Simulation Results with Weight for Reputation of 0.0

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Jobs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13.00</td>
</tr>
<tr>
<td>Simulation 1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>Simulation 2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>Simulation 3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>Simulation 6</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13.00</td>
</tr>
</tbody>
</table>
In the curve SLA violation vs. Weight for reputation shows that the number of SLA violations is reduced when the weight for reputation value is increased. This result gives evidences to claim that the use of reputation metrics considerably improves the resource allocation in market-based Clouds. The curve is shown in the following figure:

![Figure 24 Curve of Posted Offer Auction with Reputation Metrics](image)

### 8.3.2 Scenario 2: Reverse Auction with Reputation Metrics

In the case of Reverse Auction, for every Weight for reputation we have run 8 simulations with 11 rounds each one, where each round takes the final Seller Agents’ reputation in the last round and resubmit the Jobs to the Cloud (Continue Simulation). This process produced a total of 88 jobs sent to the Cloud per simulation ($8 \times 11$). Then, we calculate the total of SLA violations at the end of the last round in each simulation. Finally we calculate the average of violations on each Weight for reputation in order to draw a curve SLA violation vs. Weight for reputation. A summary of the simulation results for the different Weights for reputation is presented in the following tables:

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight for Reputation</td>
<td>0.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># of Jobs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulation 1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>Simulation 2</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>Simulation 3</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>Simulation 6</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Simulation 7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Simulation 8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8.00</td>
</tr>
</tbody>
</table>

Table 9 Reverse Auction Simulation Results with Weight for Reputation of 0.99
Table 10 Reverse Auction Simulation Results with Weight for Reputation of 0.75

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Jobs</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>40</td>
<td>45</td>
<td>50</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>Simulation 1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulation 2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulation 3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>Simulation 6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Simulation 7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Simulation 8</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9.00</td>
</tr>
</tbody>
</table>

Table 11 Reverse Auction Simulation Results with Weight for Reputation of 0.5

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Jobs</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>40</td>
<td>45</td>
<td>50</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>Simulation 1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Simulation 2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>Simulation 3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>Simulation 6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>Simulation 7</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>Simulation 8</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9.13</td>
</tr>
</tbody>
</table>

Table 12 Reverse Auction Simulation Results with Weight for Reputation of 0.25

<table>
<thead>
<tr>
<th>Round</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Jobs</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>40</td>
<td>45</td>
<td>50</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>Simulation 1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>Simulation 2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>Simulation 3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>Simulation 4</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>Simulation 5</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>15</td>
</tr>
<tr>
<td>Simulation 6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>Simulation 7</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>Simulation 8</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10.50</td>
</tr>
</tbody>
</table>
The curve SLA violation vs. Weight for reputation using Reverse Action also shows that the number of SLA violations is reduced when the Weight for reputation value is increased. This result gives evidences to claim that using reputation metrics has been possible to improve the resource allocation in a market-based Cloud. The curve is shown in the following figure:

![Figure 25 Curve of Reverse Auction with Reputation Metrics](image-url)
8.3.3 Comparison between Posted Offer and Reverse Auction Curves
If we compare the two curves from the two different market mechanisms with reputation metrics is evident that the resource allocation performance in the Posted Offer is highly improved when it is set a high value to the Weight for reputation. This is caused due to Posted offer it is less price-driven than the Reverse Auction. So, the role of reputation metrics plays a more important role. However, it is remarkably that both market mechanisms are positively influenced by the reputation metrics even though they represent opposite market power approaches. The two curves are shown in the following figure:

![Curves of Posted Offer and Reverse Auction with Reputation Metrics](image)

8.4 Threats to Validity
Even though our experiments clearly indicate a favourable impact of the use of reputation metrics on resource allocation in market-based Clouds, we must reflect on the effect produced by changing one or more of our assumptions in the model. For instance, the validity of our analysis might be affected if we consider collusion among Seller Agents because some of them may cooperate to take other Seller Agent out of the business by offering prices under the production costs, which might lead to a different resource allocation result. Another change in our model assumptions that might affect the validity of our analysis is considering varying network connectivity. This way, we would need to clearly identify SLA violations caused by networks delays. Relaxing others model assumptions would have little or no effect on the results.
9 Project Management

9.1 Risk Management
We identified the risks associated with the project and the suggested strategies that could have been taken to deal with each risk.

<table>
<thead>
<tr>
<th>Risk</th>
<th>Probability</th>
<th>Impact</th>
<th>Strategy type</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wrong selection of the trading mechanisms more suitable for the model to understand them. This might lead to a project delay</td>
<td>High</td>
<td>High</td>
<td>Weekly meetings with supervisor after reading the most important trading mechanisms</td>
<td>Reduce probability</td>
</tr>
<tr>
<td>Not reaching an appropriate understanding of the CloudSim simulation approach due to the lack of documentation. This might lead to a project failure.</td>
<td>Medium</td>
<td>High</td>
<td>Contact the team of developers to solve the complex doubts</td>
<td>Reduce probability</td>
</tr>
<tr>
<td>Discover bugs in the framework which might lead to implementation delays</td>
<td>Low</td>
<td>Medium</td>
<td>Contact the team of developers to ask support</td>
<td>Impact reduction</td>
</tr>
<tr>
<td>Lack of experience in designing controlled environments for Experimental Economics</td>
<td>Medium</td>
<td>Medium</td>
<td>Check the model with the supervisor before proceeding with the implementation</td>
<td>Reduce probability</td>
</tr>
<tr>
<td>Source code corrupted</td>
<td>Low</td>
<td>High</td>
<td>Make use of subversion repository</td>
<td>Impact reduction</td>
</tr>
</tbody>
</table>

Table 14 Risk Assessment

9.2 Development Process and Schedule
The project lifecycle most suitable for our project is the Waterfall Model. In order to monitor the results, a meeting with the supervisor was held at the end of each task of the project. The project schedule is shown in the figure below:

Figure 27 Project Schedule

9.3 Backup Management
Backups of the source code were taken every day and saved in a remote repository. Moreover, all the papers, digital books and other resources taken into account for the dissertation were also included in the backups.
10 Conclusions and Future Work

In this paper, we have introduced a new model to improve the resource allocation in the Cloud that merges the concepts coming from market-based systems and reputation systems by means of establishing a relationship between price and reputation. We have implemented the model and created a controlled environment through a simulation tool. The evaluation of the results gotten from the performed simulations in this research has given an evidence to show that the reputation metrics significantly improve the market-based Cloud resource allocation. Additionally, we have compared opposite trading mechanisms with the market power centered on the Seller like Posted Offer or on the Buyer like Reverse Auction, in both cases the use of reputation metrics has reduced the number of SLA violations. Nonetheless, the inclination of the curve is higher in a less price-driven mechanism like Posted Offer than in a mechanism more influenced by price like Reverse Auction.

However, our study only considers two market mechanisms with the market power on either the buyer or server side. Future research could implement another market mechanism like Double Auction which tries to balance the power. Another aspect that might enrich the design is the inclusion of more criteria to consider them as SLA violations. For example: deadline for a job completion. Regarding the formula to establish the relationship between price and reputation, this has been a first proposal used for experimentation purposes. It might be improved by considering new factors or by using different formulas depending on other key parameters. From the experimentation point of view, further work to extend the Seller and Buyer Agent behaviors might be done to increase the complexity of the scenarios. Finally, the tool might be extended to be used for Service Composition for Federation of Clouds.
Bibliography

22. Lin, W., Lin, G. and Wei, H. Dynamic Auction Mechanism for Cloud Resource Allocation: National Taiwan University.
## Appendix

### Appendix A: Test Case Specification

<table>
<thead>
<tr>
<th>Test case code</th>
<th>Input(s)</th>
<th>Expected output(s)</th>
<th>Actual output(s)</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC-001</td>
<td>In the Sellers tab, select a row and then press Add Default One button</td>
<td>A new default Seller Agent type one should be added at the end of the table of Seller Agents.</td>
<td>A new Seller Agent type one was added at the end of the table of Seller Agents</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-002</td>
<td>In the Sellers tab, select a row and then press Add Default Two button</td>
<td>A new default Seller Agent type two should be added at the end of the table of Seller Agents.</td>
<td>A new Seller Agent type two was added at the end of the table of Seller Agents</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-003</td>
<td>In the Sellers tab, select a row, modify the minimum profit of the selected Seller Agent to 2 and then press Save button</td>
<td>The field validate the new value and as it is out of range is not allowed and show the following error message when Save button is pressed: “Minimum profit must have a value”</td>
<td>The new value cannot be entered and the error message “Minimum profit must have a value” was shown after pressing the Save button</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-004</td>
<td>In the Sellers tab, select a row and then modify the minimum profit of the selected Seller Agent to 0.7 and then press Save</td>
<td>The field should validate that it is in the permitted range and update the minimum profit.</td>
<td>The updated minimum profit was displayed in the corresponding row of the table</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-005</td>
<td>In the Sellers tab, select a Seller Agent and press Delete from the list containing more than one Seller Agent</td>
<td>The selected Seller Agent should be removed from the list</td>
<td>The Seller Agent was removed from the list</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-006</td>
<td>In the Sellers tab, select a Seller Agent and press Delete when the list only contains that Seller Agent</td>
<td>The Seller Agent should be removed from the list and must display a message notifying that the scenario must contains at least one Seller Agent</td>
<td>The Seller Agent was not removed and the error message was displayed</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-007</td>
<td>In the Trading tab, select Both trading mechanisms and configure their weight for reputation as 1.5</td>
<td>The weight for reputation values will be validated and as they are out of range, the fields will not allow to set the weight for reputation as 1.5</td>
<td>The new weight for reputation values were not allowed</td>
<td>Pass</td>
</tr>
</tbody>
</table>

*Table 15 Test Case Specification in Accordance with IEEE Standard 829-1998*
<table>
<thead>
<tr>
<th>Test case code</th>
<th>Input(s)</th>
<th>Expected output(s)</th>
<th>Actual output(s)</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC-008</td>
<td>In the Trading tab, select Both trading mechanisms and configure their weight for reputation as 0.5</td>
<td>The weight for reputation values will be validated and as they are inside of range, the fields will be set as 0.5</td>
<td>The new weight for reputation values were allowed</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-009</td>
<td>In the Scheduled Jobs tab, select a row and then press Add Default One button</td>
<td>A new group of scheduled jobs type one should appear at the end of the table of scheduled jobs</td>
<td>A new group of scheduled jobs type one was added at the end of the table of scheduled jobs</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-010</td>
<td>In the Scheduled Jobs tab, select a row and then press Add Default Two button</td>
<td>A new group of scheduled jobs type two should appear at the end of the table of scheduled jobs</td>
<td>A new group of scheduled jobs type two was added at the end of the table of scheduled jobs</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-011</td>
<td>In the Scheduled Jobs tab, select a row, modify its scheduled time to 10 and then press Save button</td>
<td>The new scheduled time should be displayed in the corresponding row of the table of Scheduled Jobs</td>
<td>The new scheduled time was displayed in the corresponding row of the table of Scheduled Jobs</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-012</td>
<td>In the Scheduled Jobs tab, select a row and modify its maximum profit to 2 and then press Save button</td>
<td>The field validate the new value and as it is out of range is not allowed and show the following error message when Save button is pressed: “Maximum profit must have a value”</td>
<td>The new value cannot be entered and the error message “Maximum profit must have a value” was shown after pressing the Save button</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-013</td>
<td>In the Scheduled Jobs tab, when the table contains more than one row, select one of them and press Delete button</td>
<td>The selected row should be removed from the table of Scheduled Jobs</td>
<td>The selected row was removed from the table of Scheduled Jobs</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-014</td>
<td>In the Scheduled Jobs tab, when the table contains only one row, select the only row and press Delete button</td>
<td>The selected row should not be removed from the list and the following error message should be displayed: “At least one group of jobs is required to start the simulation”</td>
<td>The selected row was not removed and the following error message appear: “At least one group of jobs is required to start the simulation”</td>
<td>Pass</td>
</tr>
</tbody>
</table>

*Table 16 Test Case Specification in Accordance with IEEE Standard 829-1998 (Continue)*
<table>
<thead>
<tr>
<th>Test case code</th>
<th>Input(s)</th>
<th>Expected output(s)</th>
<th>Actual output(s)</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC-015</td>
<td>From the toolbar, press Run Simulation button</td>
<td>The system should perform the simulation, activate the Continue Simulation button and then make two new tabs appear at the end: one displaying the pie and bar charts, and the other tab displaying the log of events</td>
<td>The system performed the simulation, activated the Continue Simulation button and then made two new tabs appear at the end: one displaying the pie and bar charts, and the other tab displaying the log of events</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-016</td>
<td>From the toolbar, press Continue Simulation button</td>
<td>The system should perform the simulation taking the Seller Agents’ reputation from the last simulation and then two new tabs should appear at the end: one displaying the pie and bar charts, and the other tab displaying the log of events</td>
<td>The system performed the simulation using the Seller Agents’ reputation from the last simulation and then two new tabs appeared at the end: one displaying the pie and bar charts, and the other tab displaying the log of events</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-017</td>
<td>From the toolbar, press Close Results button</td>
<td>The system should display ask for a confirmation with the message: “Are you sure that you want to close Chart and Log tabs?” and after the confirmation it should close all the Chart and Log tabs</td>
<td>The system asked for a confirmation with the message: “Are you sure that you want to close Chart and Log tabs?” and after the confirmation was positive it closed all the Chart and Log tabs</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-018</td>
<td>From the menu File, choose Export and then choose a file name and its location to save the exported scenario</td>
<td>The system should create a file or replace an existing one with the scenario exported in a XML format in the given location.</td>
<td>The system saved a file with the scenario exported in a XML format in the given location.</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-019</td>
<td>From the menu File, choose Import</td>
<td>The system should open the specified file, load the exported scenario to the simulator and display the message “Import Completed”.</td>
<td>The system opened the specified file, loaded the exported scenario to the simulator and displayed the message “Import Completed”.</td>
<td>Pass</td>
</tr>
</tbody>
</table>

Table 17 Test Case Specification in Accordance with IEEE Standard 829-1998 (Continue)
<table>
<thead>
<tr>
<th>Test case code</th>
<th>Input(s)</th>
<th>Expected output(s)</th>
<th>Actual output(s)</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC-020</td>
<td>From the menu File, choose Exit</td>
<td>The system should ask for a confirmation to exit with the following message “Are you sure to exit?” then with a positive confirmation, the simulator should be closed</td>
<td>The system asked for a confirmation to exit with the following message “Are you sure to exit?” then with the positive confirmation, the simulator was closed</td>
<td>Pass</td>
</tr>
<tr>
<td>TC-021</td>
<td>From the menu Help, choose About Simulator</td>
<td>The system should display a dialog containing basic information about the simulator</td>
<td>The system displayed a dialog containing basic information about the simulator</td>
<td>Pass</td>
</tr>
</tbody>
</table>

Table 18 Test Case Specification in Accordance with IEEE Standard 829-1998 (Continue)
Appendix B: Other Simulator Screenshots

Figure 28 Screenshot of Importing / Exporting Menu

Figure 29 Screenshot of Exporting a Scenario
Figure 30 Screenshot of Importing a Scenario

Figure 31 Screenshot of About Simulator Dialog
Appendix C: Instructions to Run the Tool

To execute the simulator using the jar file:

Pre-requisites:

1. JDK 1.7.0 or above

Steps:

1. In the CD, go to the folder “simulator”
2. Run the command java -jar marketbasedcloud1.0.jar (libraries are in the lib folder)

To execute the simulator from the source code

Pre-requisites:

1. JDK 1.7.0 or above
2. Recommended NetBeans IDE 7.0.1

Main class: uk.ac.bham.cs.simulation.tool.gui.SimulationTool

Project Name: marketbasedcloud1.0

Steps:

1. In the CD, go to the folder “sourcecode”
2. Open the project and configure the libraries in the lib folder
3. Run the project using the main class
Appendix D: Example of XML Export File of an Scenario

<?xml version="1.0" encoding="UTF-8"?>

<scenario>
  <sellers>
    <sellerAgent>
      <name>SellerAgentA</name>
      <initialReputation>250</initialReputation>
      <minimumProfit>0.4</minimumProfit>
      <preferredProfit>0.35</preferredProfit>
      <mipsPerCore>25</mipsPerCore>
      <probabilityToSucceed>0.95</probabilityToSucceed>
      <region>NORTH</region>
    </sellerAgent>
    <sellerAgent>
      <name>SellerAgentB</name>
      <initialReputation>250</initialReputation>
      <minimumProfit>0.4</minimumProfit>
      <preferredProfit>0.25</preferredProfit>
      <mipsPerCore>25</mipsPerCore>
      <probabilityToSucceed>0.8</probabilityToSucceed>
      <region>NORTH</region>
    </sellerAgent>
  </sellers>
  <trading>
    <marketMechanism>
      <type>Posted Offer</type>
      <weightForReputation>0.3</weightForReputation>
    </marketMechanism>
  </trading>
</scenario>
<iterations>
  <iteration>
    <id>1</id>
    <numberOfUsers>1</numberOfUsers>
    <scheduledTime>0.0</scheduledTime>
    <miPerJob>1000</miPerJob>
    <mipsPerVm>250</mipsPerVm>
    <minimumReputation>1</minimumReputation>
    <maximumProfit>0.39</maximumProfit>
    <region>NORTH</region>
    <priority>HIGH</priority>
    <reliability>HIGH</reliability>
  </iteration>
  <iteration>
    <id>2</id>
    <numberOfUsers>1</numberOfUsers>
    <scheduledTime>0.0</scheduledTime>
    <miPerJob>1000</miPerJob>
    <mipsPerVm>250</mipsPerVm>
    <minimumReputation>1</minimumReputation>
    <maximumProfit>0.39</maximumProfit>
    <region>NORTH</region>
    <priority>HIGH</priority>
    <reliability>HIGH</reliability>
  </iteration>
</iterations>
</scenario>