Using Service Level Agreements to Promote Service Aggregation in P2P Networks

Ioan PETRI
Business Information Systems Dept
Babes-Bolyai University of Cluj-Napoca
ioan.petri@econ.ubbcluj.ro

The increasing intensity of resource traffic within distributed environments requires powerful instruments for controlling the system. Accessing computing resources can have varying requirements in accordance with specific user parameters and profiles. Service Level Agreements have been validated by the research community as being useful instruments for regulating the exchange of resources. Service level agreements (SLAs) can play an important role, since they are designed as contracts to determine the price for a service at an agreed quality level as well as the penalties in case of SLA violation. An SLA contains guarantee terms that need to be satisfied by a provider, and a payment that needs to be made by a user when such guarantees have been met. In this paper we make use by Service Level Agreements in order to promote service allocation. We use a computer simulation approach and investigate how local interaction of peer nodes, endowed with specific service capabilities and individual behaviors, may produce an aggregate network structure where service requests are differentiated based on their utility.
Keywords: *SLA, aggregation, P2P networks*

1 Introduction and related work

Computation on decentralized environments requires scalability and fault tolerance in order to ensure anonymous communications and resource sharing. Because of the needed mechanisms to locate nodes inside architecture, aggregation policies became methods to enhance peers collaboration. Multiple interactions between nodes system require corresponding policies to oversee resources discovering, negotiation and collaborative operations. To operate and coordinate services within a dynamic environment built by various computer applications requires a highly controlled set of policies. The central entity needs to authorize what is operated, who is allowed to operate, and the conditions under which the operations occur.

Inside the peer to peer architecture an node acts as a server for the other node as well as it can be a client requesting another node service. In the system proposed by Homayounfar et. al [1] the advanced P2P concept evolves from the level of being a simple solution for using nodes' ability to perform and optimize user operation to one level of complex answers that solve cooperation on virtual environments.

The nodes are splitting one problem received from a user or from another node and call on necessary services. Tasks are allocated for each available service node inside a negotiation process; the final solution is built by merging all the sub-solutions.

On Gnutella [5] it is develop a decentralized environment where the nodes are gathered on group membership. Such architecture allows users to join or leave frequently and the major involved operation is sharing. The number of nodes increases the sharing space and files availability with a constant response time. Nodes can join the network by connecting to other available hosts. After joining the network they broadcast messages in order to interact one with each other.

In scientific collaboration node [6], grouping is requested by many reasons: sharing interest, working for the same company, geographical proximity. One node interaction is required by the proximity in order to exchange information without having to pass through any dedicating server. On member node can join the virtual organization only if

it is invited. Each node should join a group and each of it has a list with all the groups from the community. The processes that one particularly member node can perform are: joining, replicating and searching processes. Every disconnection of one member node displays a message of no reply for the network status.

A service provider approach is developed on [7] showing that a provider needs authentication in order to deliver a service for a particular community. The provider node must accomplish all the community policies and to relate all its particular policies to the community requirements. An agreement framework is developed here between any new node that wants to deliver a service to another node from a different community or to the community itself. The community may require certificate for authentication in order to allow the node to submit the service to the community. Thus, from the community prospective authorization represents a mandatory commitment for every external service-node. According to its evaluation one node can receive rights described in capabilities to broadcast services inside the community.

This paper is about to investigate the problem of aggregation and the policies of controlling the environment. Equally, we investigate the business implications translated in financial clauses that are established between various participants.

Services also have quality parameters that influence the agreement process. Eventually, negotiation can be the process that creates the agreement framework between entities. Policies and objectives between provider and requester produce particular quality of service with an established time parameter. [8]

2 Background and approach

As the new concepts of computation are focused on distributed environments, many of important research orientations decide to use a distributed framework. Inside the distributed environment every node is a peer (node) that can serve others or it can be a service for a particular purpose. The services are distributed on nodes, each node being considered an autonomous entity. The considered infrastructure for deployment is a P2P SOA pervasive network where nodes behave as service consumers and producers. Any service represents a case of an interactional process that can involve data exchange.

The term of service defines a special resource that has the capability to perform tasks with a coherent functionality (provider nodes and requester). Inside the P2P architecture a service is considered to be a resource that is transferred between two or more entities. At the same time one service performs one or more tasks, it has a services description, a service interface, service semantics, and an identifier.

One service has one or more roles in relation to service owner. Policies that are applied to service are highly significant for its definition [3]. Thus, on our default infrastructure several mechanisms as node preferences, production of social welfare, complexity, negotiation and algorithm, appear in order to extend a basic unstructured P2P networks with autonomous nodes.

On such architecture, each node is supposed to deliver or receive services from other nodes. Nodes can operate different services on the network. Particularly, we develop the research on a set of N nodes that operate services in an unstructured P2P architecture. We consider that every p node has a q service quantity and it is linked with other nodes building virtual organizations. The service S that one node owns can contain a set of attributes (b_1, b_2, b_3) that characterized the service. These attributes are the price of service, the quality of service, time the service operates, etc.

We use also a set of indicators as the expected value for the delivered service and the requested price for the delivered service. According to figure 1 every initiator is able to discover available services that can be operated on the network as well as it can have a list of possible service providers from the network.

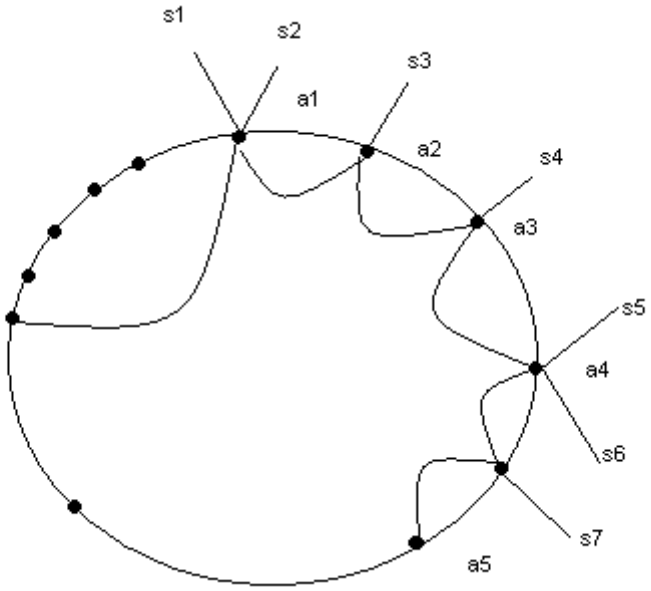

Fig. 1. A SOA peer to peer network

Services can be uniform distributed among the nodes or some node may have different quantity of services at different prices. Basically one service can be expensive than others and the quantity of services are changing from an node to another. The service delivery is a process that involves time (Any *T* unit time added to a service request indicates the affected period of time that an initiator node will use the service in operations. Any provider node involved on a service transaction, will avoid system errors instead of committing the service to another requester).

The process starts from an initiator that requires service on the network. The requested service has *Q* quantity and *T* time parameters to be delivered. The initiator node builds a list with the entire available services from the network and all their particular service parameters values. A service level agreement framework will be developed in order to operate service transaction. [2]

3 Negotiation and trust for collaborative entities

Negotiation is a decision factor for reaching a good agreement. In the agreement process nodes can have preferences, they can use privacy and reliability as well as they can aim to reach an agreement under specific established circumstances.

Services are requested at different levels, either as resources at a local level (node from the same virtual organization) or as resources at a global level (from another virtual organization). Services are objects of the negotiation process being consumed or provided at the same time. Any service is specified by different negotiable parameters: type and amount of service, the time period a service is used, the reward and the penalty. The final result of the negotiation is considered a contract between the node entities accompanied by all the negotiation parameters. An agreement is meant to solve a conflict of interest after a sequence of determined steps while different node strategies are applied.

Instead of negotiation, the authority criteria can be introduced. The level of authority can change the negotiation of services into an authority based framework. One node requiring a service can have priority if the level of authority is higher than the authority level of the opponent interaction node. [10]

After the service node gathers enough details about the potential client it can establish all the parameters involved in the service operation (delivery or request). For the peer-to-peer infrastructure, reputation can allow nodes to cooperate each other. Reputation represents a concentrated stored history of activities averaged with transactions

information, time and transaction values. Any external node joining an organization can receive a trust value based on a pre-existing relation. A reputation model can evolve from a level to another by changing the history and new reputation rules [11]. The trusting behavior is reflected in the reputation, therefore new events and behaviors derive from existing reputation. Any new event and action is added to the history for building a database for recommendations. After each operation the reputation information is updated.

Reputation mechanism introduces trustworthy behaviors with trustful ratings for each node from the organization. These reputation ratings are influencing the node's behaviors in the process of negotiation.

Our central subject is developing among the costs implication ignoring the problem of reputation and identity. This paper focuses on a business perspective of the system statuses that can be influenced by different configurations of the environment.

4 Service agreement collaboration

Previously, we presented several aspects for a better understanding of the approach. The agreements on virtual organizations involve a specific infrastructure in order to benefit of accurate negotiation mechanisms. Service level Agreements (SLAs) are dynamically establishing policies between two collaborative entities .The objective of the interaction is to deliver a service to a requester. This delivery process involves functional and non functional properties for the service. The collaborative entity requires a management of the delivery process using rights, roles and obligations of the involved nodes as presented in figure 2.

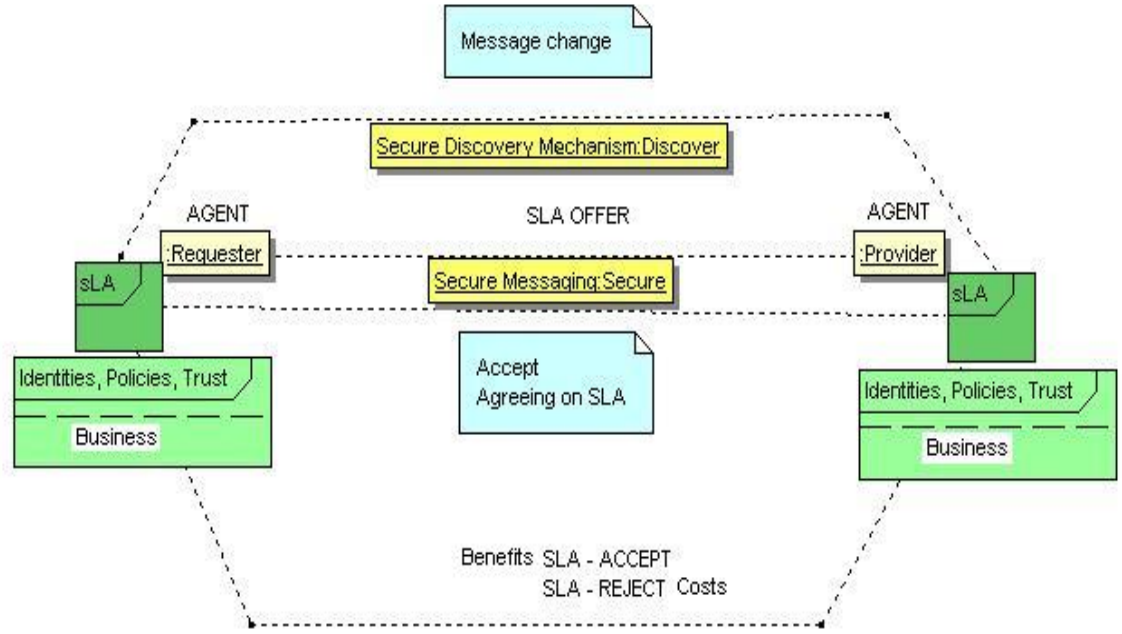

Fig. 2. Service Level Agreements

A set of constraints are established in this representation:

- Purposes reflecting reasons behind the creation of the SLA
- Parties designing the nodes involved to SLA and their perspective roles(provider and requester)
- Validity period evaluating the period of time that the SLA will cover
- Scope reflects services covered in the agreement
- Restrictions as constrains, in the form of possible work-flows, needed to be followed for the requester service

One requester makes a SLA offer for the provider The offer uses a sets of parameters that the provider can analyze. The agreements between providers and requesters

imply restrictions specifying terms of non-performing objectives (any service delivered to unpredicted agreed status outcomes a penalty), optional services (delivered services that are not required by requester), exclusion issues (are not included to SLA but might be required as an exception), administration processes (for meeting and measuring SLA objectives and defines organizational responsibility for those processes. Service level agreement clauses have more familiar forms when they contain the whole process). Requester can build SLAs for different consumers as well as one consumer can have many SLAs for different requesters. This process is considered as a different behavior dividing across multiple providers (work-flow composition).

Inside a complex scenario SLA clauses can bring dependencies. Co-allocation is thought to be any SLA validation bound by a different SLA agreement.

At the forming agreement process it should be distinguished between agreements and the mechanism that leads to the formation of the agreement. One negotiation is effected by the level of SLA and the expected value of the agreement process. After the agreement has been signed the nodes enact the contents of the agreement clauses with different degrees of success. For a very accurate process, the agreement level defines the following steeps:

- Identity provider managed using descriptors
- SLA definition of what is being requested
- Agree on SLA terms requires definition of the SLA parameter Service Level Objectives(SLO)
- Monitor SLA Violation as an internal process of SLOs recognition
- Destroy SLAs with expired timestamps
- Penalty for SLA violation reflects a non meeting attitude of expectation

An extended approach of service collaboration involves identical servers for executing dynamic allocation among multiple web applications. For each application they have a performance optimizer application manager interacting with a resource arbiter (server allocation) using optimization on the objective function (expected business value).

The resource arbiter allocates resources for maximizing the sum of the expected business value over all applications. It is important to note that the local value functions must share common scale. [15]

For SLA clauses parameters [15] defines:

- Service Level Objective representing the objective that the service should manage and the level of quality that the service should satisfy (SLO).
- Service Level Indicator, values parameters of the SLA clause. They are expected values that had to be followed during the interaction process (SLI). They can be integers or decimals values as well as they can be considered constrains.
- Business Value as a value returning function (level of business build through its reputation)(BV).
- Penalties that represent the most important element of the SLA clause. The systems can oversee the whole interaction mechanism by using penalties (Pen).

SLA clauses violations are considered unexpected and deviated behaviors in completing the SLA objectives. It is very important to study the impact of SLOs (Service Level Objective) on the penalty dimension. Due to its dimension, a deviated behavior can involve either penalty or rewards (they represent important elements for the business use of SLAs). The interaction between requesters and providers can experiment different levels of SLAs violations: Completion "all or nothing", Completion "Partial", Completion "Weight partial".

Every SLA clause is built to be completed with any amount of utility bigger than 90 percent of the required service. For an level of provisioning less than 90 percent of the required service, the penalty can grow with 1% monetary units for each uncompleted percent.

5 Simulations and protocol

This paper uses an simulation based

approach in order to investigate several behavioral characteristics of the system. The simulation represents a fundamental element of the research design. It helps to validate modeled assumption in the context of a high complexity. We investigate how various peer nodes holding a specific service distribution can deliver services on the basis of SLA establishment among parties. The simulation is aimed to confirm different hypothesis that can address the problem of service aggregation in a peer-to-peer context.

On our framework one peer node can play the role of provider or client. The provider receives a service request from one client and offers an SLA template as an answer to the request. Each peer is modeled as an entity that can deliver several services with specific attributes. The entities of the distributed systems represent intelligent actors from a network. They can solve problem from other nodes, they can discover needed resources, and they can negotiate with other nodes and offer solutions. Nodes can also learn, update and predict operations as well as they use the resources located on each other optimally and work as a team.

Our simulations are carried out using the PeerSim simulator [16] that offers a common platform for research projects on P2P networks. It is an open source, Java-based simulation framework for developing and testing P2P algorithms in a dynamic environment. PeerSim has been designed to be both dynamic and scalable and uses a simple ASCII file based configuration mechanism. PeerSim enables different modules to be developed that encode various capabilities required within an application. PeerSim also provides pre-developed modules for building and initializing an underlying network, and modules to control different interaction protocols between peers. To support demand manipulation, where new requesting peers need to be added to the peer-to-peer system, the environment must be designed as a large dynamic network.

Simulation in PeerSim is controlled through Java objects that can be scheduled for execution at certain points. These controllers typically initialize, observe or modify the simulation. The initialization phase is carried out by control objects that are scheduled to run only at the beginning of each experiment. In the configuration file, the initialization is undertaken using a Scheduler object which identifies the simulation cycles, and when they are executed exactly. This object can also be used to configure a protocol or be executed in specific simulation cycles. It is also possible to control the order of the running of the components within each cycle. The model dissociates the participants according to their configured profiles. Therefore, one node can be a provider as entities holding a specific service distribution that can be deliver to other requesting entities (peer nodes) or one node can be a client as a requesting entity that needs specific services in order to perform different internal tasks.

The framework uses a P2P network in which each node has a specific number of links that is specified in the configuration file. One node can deliver a specific number of sub-services n = {1,2,3,4,5} that can be varies according with the simulation context. Each node is assigned with an *utility* value that is calculated at each simulation cycle and an accepted price *Acc_price* representing the minimum income that one peer would accept in order to deliver a specific sub-service.

Each simulation process uses a specific number of cycles, each cycle having assigned a number of effects to take place. One node receives a service request which must be solved by aggregating sub-services from their immediate neighbors.

The process takes place within the following steps as depicted in figure 3:

- One sub-provider *i* runs an internal algorithm for verifying its capability to satisfy the request.
- One provider performs a search algorithm to check which of its immediate neighbors can work on the aggregation (can satisfy the SLA)

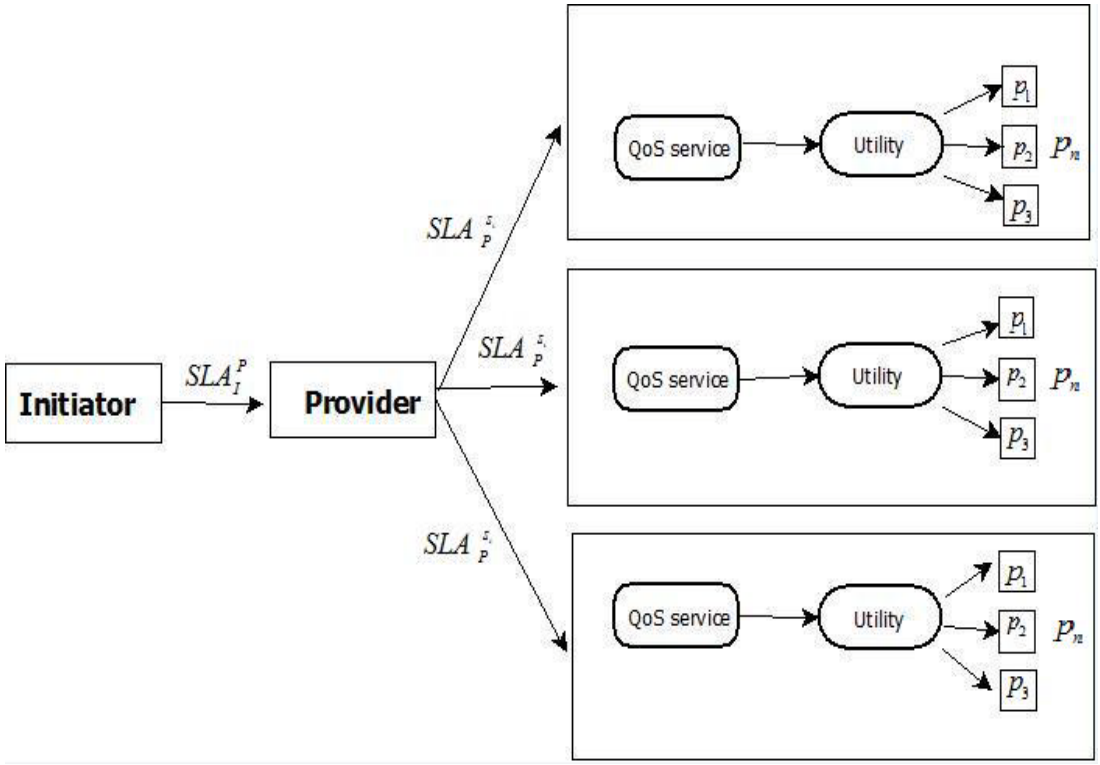

Fig. 3. The process of aggregation

- At the moment when the SLA ends the monetary payments are distributed among participants.

6 Results

Each simulation is executed within a framework of 10000 peer-nodes. Each experiment is using 100 simulation cycles to expose the behavior of the system in the context of different configurations files.

Experiment 1 investigates the number of completed operation when the system uses a varying number of configured sub-services.

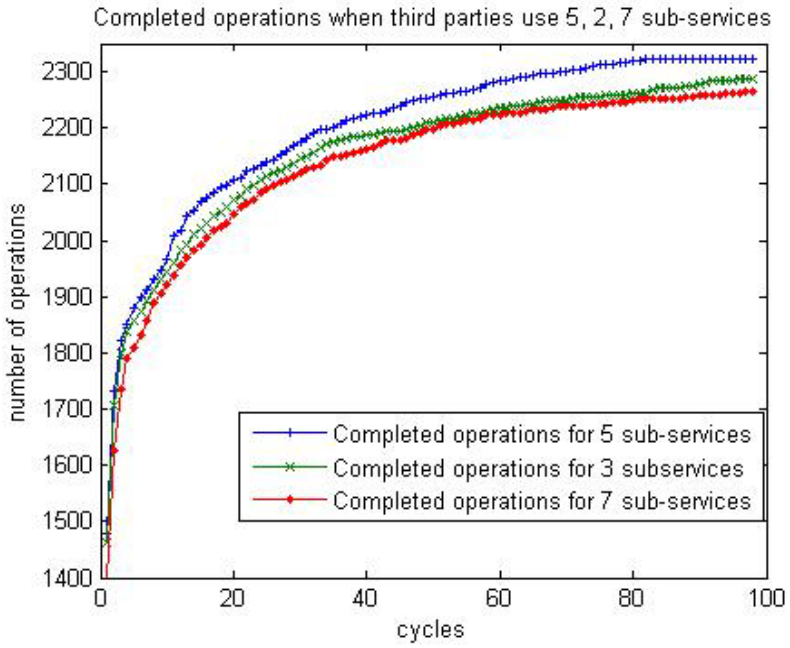

Fig. 4. Completed operations with different number of configured sub-services

As fig. 4 illustrates, in the context of 5 sub-services to be aggregated the system records

the highest number of completed operations. It is also can be observed that a number of 7 sub-services to be aggregated induce the lowest level of completed operations. We can conclude that 5 sub-services is an optimal value for the system. All the curves become stable after 80 execution cycles.

Experiment 2 presents the classification of the payment rates in the context of a varying number of sub-services to be aggregated.

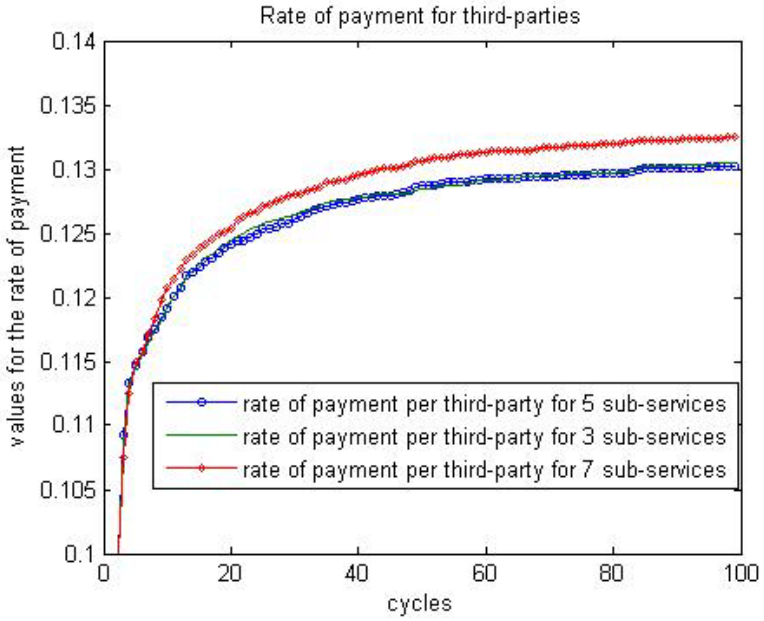

Fig. 5. Rate of payment per sub-service

Figure 5 confirms that after the completion of 100 simulation cycles the system records the highest level of payment rates in the context of 7 aggregated sub-services. The configuration with 3, respectively 5 sub-services produce approximately the same level of payment rate.

Experiment 3 presents the status of the system measured in term of completed operation in the context of two different configurations. The number of participant used for simulation is varied from 10000 participants to 5000 participants.

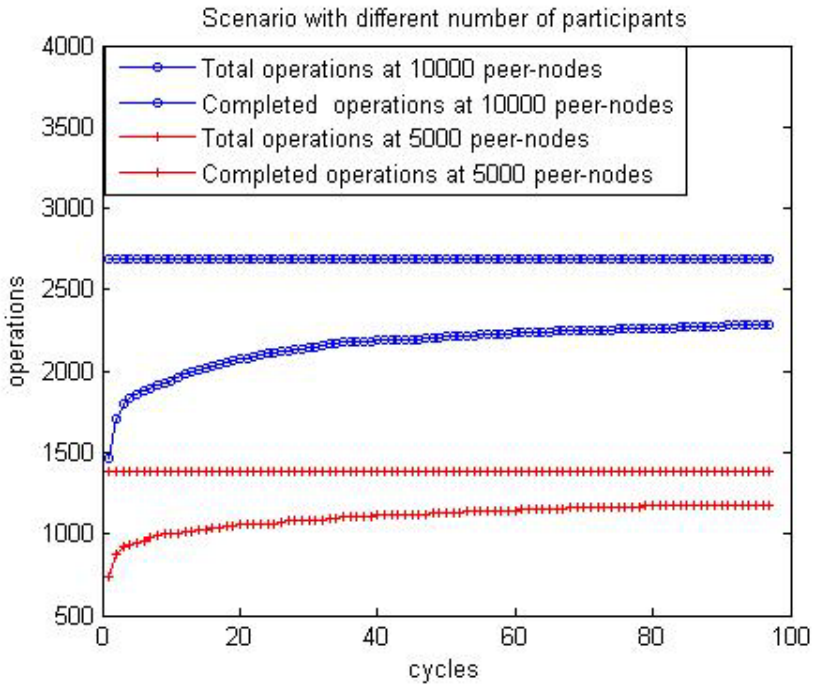

Fig. 6. The completed operation with different numbers of participants

From figure 6 it can be observed that the size of the network (the number of participants) is a decisive factor that can induce different behaviors of the system. In the context of 10000 participant the number of the completed operation is much higher that for 5000 participants.

7 Conclusions

This paper present with a simulation based approach a model for the service aggregation. We explained how one provider node can aggregate a different number of sub-services (see figure 6) and we investigate how the value of different system parameters (completed operations, rates of payment) changes according with specific configurations.

Various models have been deployed in order to solve the service interactions in distributed environments. Our target was to build a research infrastructure that solves the puzzle around service aggregation where financial parameters can induce a desirable framework for negotiation. We use Service Level Agreements (SLAs) as mechanisms to reduce conflicts appearance inside the agreement framework. Further we present a simulation framework in order to clarify particular situations when a negotiation ends with financial incentives or penalties.

Acknowledgements

This work is supported by the Romanian Authority for Scientific Research under contract IDEI 573

References

[1] H. Homayounfar, F. Wang and S. Areibi, "Advanced P2P Architecture Using Autonomous Nodes," *International Journal of Computers, Systems and Signals*, 2002.

[2] M. Moca and G. C. Silaghi, "Resource Aggregation Effectiveness in Peer-to-Peer Architectures," *Advances Grid and Pervasive Computing*, Berlin, 2009.

[3] D. K. Barry, *Web Services and Service-Oriented Architecture: The Savvy Manager''s Guide, Edition Illustrated*, Published by Morgan Kaufmann, 2003.

[4] L. Pearlman, V. Welch, I. Foster, C. Kesselman and S. Tuecke, "A Community Authorization Service for Group Collaboration," *IEEE 3rd International Workshop on Policies for Distributed Systems and Networks*, 2002.

[5] M. Ripeanu, "Peer-to-Peer Architecture

Case Study: Gnutella Network," *Department of Computer Science*, University of Chicago, 2001.

[6] J. Mitre and L. Navarro-Moldes, "P2P Architecture for Scientific Collaboration," *Polytechnic University of Catalonia*, Spain, 2008.

[7] L. Pearlman, V. Welch, I. Foster and C. Kesselman, "A Community Authorization Service for Group Collaboration, Policies for Distributed Systems and Networks," *Proceedings of Third International Workshop on Policies for Distributed Systems and Networks*, 2002.

[8] J. Li, K. Mong Sim and R. Yahyapour, "Negociation Strategies Considering Opportunity Function for Grid Scheduling, Euro-Par 2007, Parallel Processing," *13th International Euro-Par Conference, Lecture Notes in Computer Science 4641*, pp. 447-456, Springer, 2007.

[9] D. R. Sarvapali, "Multi-Node Negotiation using Trust and Persuasion, Euro-Par 2007, Parallel Processing," *13th International Euro-Par Conference*, 2007.

[10] C. Sierra, N. R. Jennings, P. Noriega, and S. Parsons, "A framework for argumentation-based negotiation," In M. Singh, A. Rao, and M. Wooldridge, Intelligent Node IV: *4th International Workshop on Node Theories, Architectures and Languages (ATAL 1997)*, Vol. 1365 of Lecture Notes in Computer Science, 1998.

[11] I. L. Telesca and F. Botto, "An Open Distributed Identity and Trust Management Approach for Digital Community Ecosystems, Open distributed System for Communities," *Parallel Processing, 13th International Euro-Par Conference*, Euro-Par 2007.

[12] M. He, N. R. Jennings and H. Leung, "On node-mediated electronic commerce," *IEEE Trans on Knowledge and Data Engineering*, 2003.

[13] S. Poslad, M. Calisti and P. Charlton, "Specifying standard security mechanisms in multi-node systems," In R. Falcone, S.Barber, L. Korba, and M. Singh editors, *Lecture Notes in Computer Science*, Vol. 2631, 2003.

[14] J. Rodriquez and J. Klug, "Federated Identity Patterns in a Service Oriented World," *The Architecture Journal*, 2008.

[15] P. Van Roy, "Self Management and the Future of Software Design," *Department of Computing Science and Engineering Universite catholique de Louvain*, Louvain-la-Neuve, Belgium, 2006.

[16] M. Jelasity, A. Montresor, G. P. Jesi and S. Voulgaris, "The Peersim Simulator," Available at: http://peersim.sf.net.

Ioan PETRI has graduated the Faculty of Economics and Business Administration from Babes-Bolyai University of Cluj-Napoca in 2007. On his doctoral orientation in Service Collaborative Systems develops a collaborative architecture for supporting the exchange of services. He is interested in programming languages as well as he particularly enjoys using new Java development frameworks.