A Simulation Framework for Socially Enhanced Applications

Mirela Riveni, Hong-Linh Truong, and Schahram Dustdar

Distributed Systems Group, Vienna University of Technology
1040 Vienna, Austria
{m.riveni,truong,dustdar}@infosys.tuwien.ac.at

Abstract. The emergence of complex computational problems, which cannot be solved solely by software and require human contribution, have brought the need of designing systems that efficiently manage a mix of software-based and human-based computational resources. Simulations are very appropriate for research on these systems, because they enable testing different scenarios, while avoiding the cost and time limitations of research in real systems. In this paper, we present a framework that enables the simulation of socially-enhanced applications utilizing both software and human based resources. Our framework addresses challenges in human-software environments, such as identifying relevant comparable metrics for mixed resources, mixed-resource selection, composition and scheduling algorithms, performance monitoring and analysis etc. We show the framework's usefulness and usability through a use case.

Keywords: Simulation, socially-enhanced applications, mixed systems.

1 Introduction

With technology advancement and the ever-evolving inter-business collaborations there is a growing need and opportunity to include humans as compute units in addition to software. There is an emergence of new types of complex and unconventional problems, which cannot be (accurately) solved solely by software and thus require human contributions. Examples of such problems are language translation, object recognition in images, common-sense reasoning [10], deciphering handwriting, and incident response management. The term Human Computation [10] was coined to describe the phenomena of using people for solving these type of problems. In our work, we are interested in socially-enhanced applications which use human computation to enhance software-based services (SBS)[1]. For example, a software-based language translation application may also utilize individual human-based services (HBS) or even a crowd [2] to evaluate the results translated from SBS and/or to edit and improve the translation.

Socially-enhanced applications bring a variety of research challenges due to the inclusion of HBS. Examples of such challenges are how to select and compose mixed HBS and SBS, when and how to include HBS in a process or a workflow

[1] In this paper we use the terms service and resource interchangeably.

P. Herrero et al. (Eds.): OTM 2012 Workshops, LNCS 7567, pp. 544–553, 2012.

(human in the loop), how to monitor the performance of HBS/SBS or mixed services, and how to schedule hybrid resources. To address these issues we need to provide test environments that support applications empowered with HBS.

Although there is already a considerable body of work regarding the characterization of services provided by humans as types of services which can be used comparably and along with software based services [7,8,2], there is a lack of tools supporting the design and testing of socially-enhanced applications on hybrid environments. Most of the tools are intended only for machine based resource environments (e.g., Grids and Clouds) or only for human-based resources (e.g., crowdsourcing platforms). Moreover, unlike tools for software-based applications, it is very challenging to design those intended for testing socially-enhanced applications in a real environment. We argue that simulations are very appropriate for testing hybrid environments, because as much as conducting research on frameworks with real data by using real HBS gives solid results, it also brings difficulties such as:

- high resource cost (e.g. humans based services and usage of machines on clouds need to be paid),
- long experimentation time,
- geographical distribution of resources, and
- difficulty in getting the same results when repeating experiments.

Simulations can provide controllable environments, with which we can experiment in a fast way and free of cost. In addition, they enable reproducible results providing for selection of (near) optimal solutions and analysing a variety of cases, e.g.,enabling process composers to change their process structure,such as modifying activitites and control flows.

In this paper, we propose a simulation framework that enables the integration of human-based computing resources with software-based ones. The purpose of the framework is to provide means for addressing the above-mentioned challenges in the development and testing socially-enhanced applications. In this paper, we provide a detailed requirement analysis for the need of a simulation framework for hybrid systems, present a conceptual design of the framework, discuss implementation issues of the framework, and provide a concrete case to illustrate the usefulness of the framework.

The rest of this paper is organized as follows: Section 2 presents related work. Section 3 details requirement analysis. We describe our framework in Section 4. Section 5 discusses an example to demonstrate the framework's usability and usefulness. Section 6 concludes the paper and outlines our future work.

2 Related Work

Human Computation Related Frameworks. A framework that allows to model human capabilities under the concept of Web services is presented in [7]. This framework includes an interface editor where "job requesters" can specify the tasks that need to be solved and a middleware which among others holds

a layer responsible for dispatching requests and routing them to the selected human based service. A more specialized framework is presented in [5]. The framework presented is exclusively intended to be used for human computation. It is not aimed for hybrid-resource problem solving but for applications of crowdsourced multimedia processing and querying. Work in [8] presents a mixed system model evaluated for a language translation application. The model includes a planner which assigns resources to tasks based on requirements; it has a component that first estimates the resources to be chosen and an execution component that does coordination. A toolkit for quality management of human provided services was presented by Brenbach et al. in [1]. The toolkit enables qualification test creation and task result evaluation. The toolkit provides a simulation mode as well. All of the above-mentioned works relate to enabling human computation either in general for SOA environments or for specialized problem solving. However, we are interested in simulations which can enable controllable experiments and research on different types of socially-enhanced use cases.

Cloud/Grid Simulation Toolkits. CloudSim [4], is a framework that enables the simulation of cloud environments and application services. It provides the simulation of large scale datacenters and federated clouds as well. Entities such as cloud users, resources, virtual machines and their behavior based on different scheduling policies can be simulated and experimented with.GridSim [3] is an event-driven simulation tool for developing and testing Grid related simulations, such as grid resource management and scheduling. Both CloudSim and GridSim do not simulate HBS. However, the richness of features, with which the GridSim toolkit provides ways of specifying different types of machine-based resources and different ways of resource scheduling and management, was the motivating factor for us to integrate it into our simulation framework.

Research Based on Industry Solutions. Amazon Mechanical Turk (MTurk) is a good example for discussing research with *real* HBS enabled platforms. MTurk, apart from being used as a human-service marketplace for micro-tasks, is also being excessively used for crowdsourcing related research. Paolacci et al. [6] have made a study of the demographics of workers on MTurk and have discussed the advantages and difficulties of running experiments on this platform. What is important for us in this work is that the authors point to concerns about the validity of results in MTurk experiments, since the workers are payed little and there are no guarantees that the quality of data is the desired one. MTurk can be a good experiment tool for certain test cases related to human computation, precisely for questions such as how payments effect the quality/quantity of work. However, MTurk is not designed for running complex tasks. For instance, we can not experiment on metrics and algorithms of matching task requirements to worker capabilities. Thus, MTurk is a good example of why crowdsourced-enabled environments are difficult to experiment with, especially in relation to socially-enhanced applications: they are limited to only human computation environments, and there are cost and time related disadvantages as humans need to be paid and real work has to be carried through.

3 Requirements for Supporting Socially-Enhanced Application Simulations

There are fundamental requirements that a simulation framework for analyzing socially-enhanced applications (SEA) needs to fulfill. We identify the following ones:

Req.1 – *Support for different ways of specifying tasks*: SEAs need both SBS and HBS-related tasks but in many cases the developer should be able to indicate whether a task is designed specifically for SBS or HBS or the simulation framework should automatically select SBS or HBS for a task, depending on constraints, e.g., availability, price, response time, and quality of data. For example, depending on the price and quality, a translation task can be mapped to Google translation service or to a human based translaton service. Furthermore, depending on constraints, tasks can be mapped to individual or composite (mixed) units.

Req.2 – *Support for soft-constraint based parameter description and storage for HBS*: the developer of SEA should not deal with complex and large numbers of SBS and HBS in task assignments. For this, selecting and mapping HBS to tasks in SEA should be done automatically via controlled parameters. Therefore, various HBS related metrics should be supported, such as skill type, skill level, and trust. These metrics can be utilized not only for HBS selection and ranking algorithms but also for the development and analysis of different voting schemes in detecting adversary HBS's for example.

Req.3 – *Support for description of SBS and HBS in a unified way*: if SBS and HBS for SEA are described in a different way, the composition and inter-exchange of SBS and HBS for tasksis difficult. Therefore, the framework should provide a unifying model of describing non-functional parameters (NFPs), which can be similarly defined for both resource types, such as: availability, location, quality of results, and pricing models. This unifying model is crucial for enabling: (i) the comparison between SBS and HBS (e.g., when running task-to-resource matching algorithms, in adaptation techniques when replacing human by software or vice versa, and managing trust between HBS and SBS), and (ii) the characterization of mixed units (e.g., analyzing the performance of a mixed unit and resource interaction efficiency within a unit).

Req.4 – *Support a rich set of scheduling algorithms suitable for mixed units*: potentially there will be different ways of using SBS and HBS. In order to support the developer to find the best way to schedule tasks, various algorithms should be provided, based on, for example, client priority, task priority, workload concerns, injecting HBS (e.g. when and where to put an HBS for evaluation of SBS output). In particular, scheduling algorithms for mixed units are crucial because they have not been well developed.

Req.5 – *Support for analysis of NFPs:* several NFPs, such as price, response time, quality of data, and reliability must be monitored and analyzed in order

to support (i) performance tradeoff analysis (e.g., price and response time), (ii) structural analysis (e.g., which mixed-unit topology is best for a case, which task routing/delegation algorithms are more efficient, bottlenecks due to dependency relations, resource unit elasticity: unit expansion and shrinkage, etc), failure/misbehavior analysis (e.g. testing of different adaptation algorithms). Such analyses are crucial for the developers of SEA in evaluating the cost of their applications and in decision-making on how to use resources and how to adapt the applications.

4 The Hybrid Simulation Framework

4.1 Architectural Overview

Figure 1 shows our proposed framework. *The Resource Unit Information Manager* acts like a repository for all existing resources. This component is utilized by the *Broker* for selecting the most appropriate resource/s for a given task. *The Broker* contains (hybrid) task-to-resource matching algorithms and ranking algorithms that are utilized for resource selection and/or composition. Thus, *The Broker* is associated with Req.1 discussed in section 3, because appropriate task specification is a prerequisite for developing matching algorithms to select (and/or rank) individual resources or hybrid-resource composite units for job execution. The output of the Broker can be a single resource, a (ranked) list of individual resources, a composite unit or a (ranked) list of composite units.

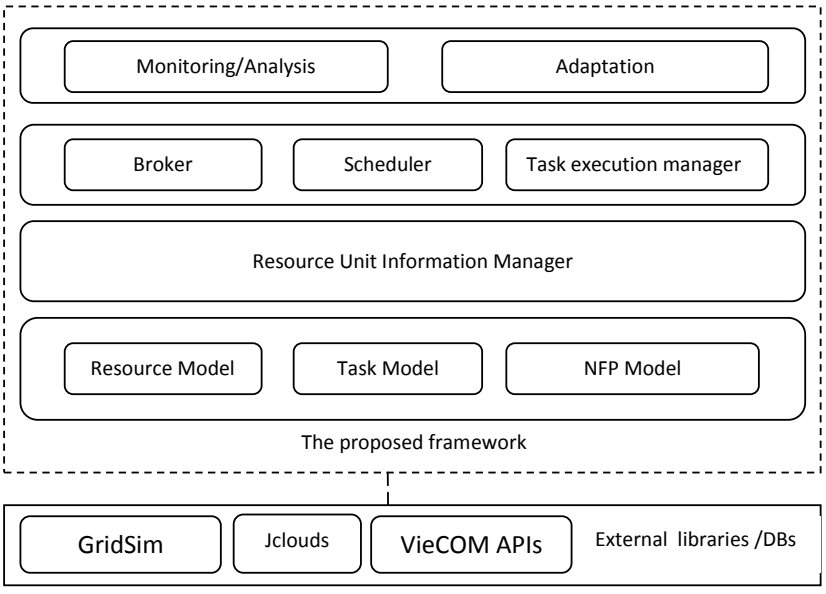

Fig. 1. The conceptual model of the proposed simulation framework

The Broker gives the *Scheduler* the list of appropriate resources for each client's tasks. The Scheduler's purpose is to enable running different scheduling algorithms (centralized/decentralized) and investigating which of them perform better for individual or composite compute units. It is our component for fulfilling Req.4. The *Execution Manager* maintains and manages the state of the application's tasks and their status. It may also be used to aggregate subtask results. The *Monitoring and Analysis* component monitors and stores information in a knowledge base about individual and composite resource performance, based on different NFPs. This component is responsible for Req.5, namely the support for NFP analysis. The *Adaptation* component utilizes information from the *Monitoring and Analysis* component, for optimization purposes. It runs adaptation and optimization functions which are used to communicate with the broker to instruct it to select new resource/s if needed, e.g., for reconfiguring the composite unit, and/or with the scheduler, to instruct it to reschedule the failed tasks. Figure 2 describes interactions among components in our framework.

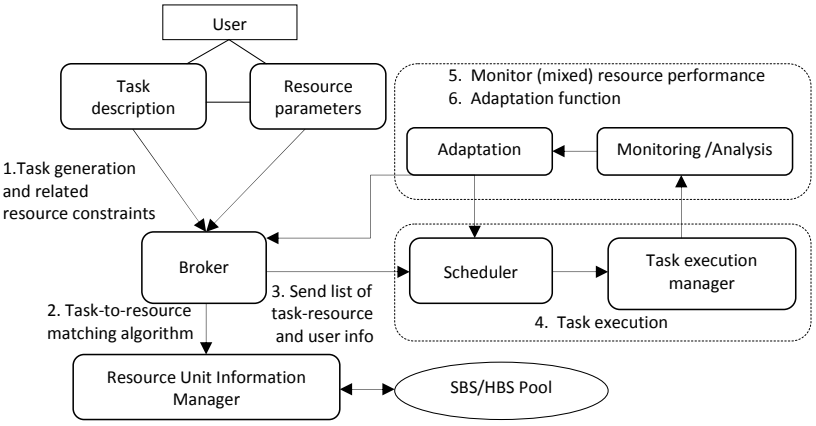

Fig. 2. Component interaction overview

4.2 Implementation

For the implementation of our framework we use the Java based GridSim toolkit[3] and build our own HBS features. Even though GridSim is primarily intended for simulation of Grid environments, several features of GridSim can be reused for modeling and executing SBS and for the development of HBS scheduling and description. Furthermore, we are integrating existing Cloud APIs, such as JClouds[2] to allow us to obtain real-world machine-based resources in different clouds. In addition, we are also integrating VieCOM APIs which are developed for accessing hybrid SBS and HBS [9].

[2] `http://www.jclouds.org`

4.3 Task and SBS/HBS Models

Preliminarily, we model both SBS and HBS task object types with the following properties: task input length, output length, price, start time, finish time, status. We model a resource by associating it with NFPs. NFPs can be static and dynamic. Table 1 describes the main NFPs in our model.

Table 1. Main NFPs for SBS and HBS

Resource Type	Static properties	Dynamic properties
SBS	Name Service Type/Architecture/OS Price Location	Availability Workload Rank Reliability Success Rate
HBS	Name Price	Availability Skill Competency Workload Reputation/Trust Score Reliability Success Rate Location

The dynamic properties are those intended for monitoring, analysis and adaptation purposes and are updated based on outcomes from SBS/HBS performance monitoring; they are functions of other properties which can be both static and dynamic. For example, the response time, can be defined as depended on the SBS and HBS capabilities respectively (e.g.,underlying resource architecture-static, human skill/competency-dynamic and updatable) workload and the tasks complexity (e.g., task input file length); availability depends on the workload and the resource's response time; the SBS/HBS success rate is defined as the number of successfully completed tasks over all assigned tasks etc. Properties such as skill, competency/skill level for HBS are responsible for satisfying Req.2, namely the soft constraints for HBS.

4.4 Scheduling Tasks

Tasks can be scheduled using a set of available algorithms (each as a plugin) for individual and composite compute units. Scheduling would take into account two-level priorities: task based and user based. This is an additional reason why the scheduler is separated from the broker. A matching algorithm can return a list of users and their associated tasks and appropriate resources, while the scheduling algorithms can be developed to take into account task and user priorities (among other factors). The design of our framework allows the integration of SBS-related, HBS-related and mixed-units related scheduling algorithms.

Appart from resource lock-in task scheduling algorithms, we are interested in developing algorithms which could enable SBS/HBS task delegation and routing capabilities even after a task is already assigned to them, e.g., task stealing algorithms.

4.5 Monitoring

When tasks are executed, monitoring events will be captured from the *Task Execution Manager* in order to determine NFPs, such as response time and failure. The Task Execution Manager will store the Task ID, their requirements, the resources to which the tasks are assigned and periodically record the task state. This state information will be sent to the Monitor. The monitoring algorithm will periodically check resource states so that if a resource has failed it can send this information to an adaptation algorithm. In addition, output from monitoring events can be used for storing feedback information from both clients and resources. Some usage examples of this feedback information include scenarios of rating and trust-based analysis. The knowledge base can be additionally used for building refined resource-selection algorithms, so they include not only resource properties but also historical performance data. For example, if there is a composite task executed by a composite unit, after job completion, a voting strategy can be implemented so that each HBS in the composition can give feedback, about the cooperation efficiency within it. This data, that shows the motivation of an individual unit to work within the same composition again, can be used as an additional efficiency or team-based trust metric for a composite-unit selection algorithm.

5 Discussion on the Framework's Usefulness

In this section, we describe a scenario that illustrates the usefulness of the simulation framework we propose. Figure 3 describes a case that shows the use of simulation to create and analyze what-if scenarios of injecting humans for evaluating quality of data (QoD) in different steps of a scientific-simulation workflow. Some of the reasons why we need HBS in long-running large resource-consuming scientific-simulation workflows are: 1) intermediate quality of result (QoR) evaluation and analysis, so as to assess if the output of SBS in a certain step is correct and appropriate to be used as input for the next step; 2) QoR enhancement by editing intermediate results (depending on the type of the simulation); 3) final QoR evaluation and analysis to check if the simulation result makes sense or not. These types of human computation are needed for enhancing the workflow execution and for optimizing the cost of compute resources, storage and people spent in long-running simulations. For example depending on the HBS evaluation, tasks in the workflow can be restarted on the same resource, stopped altogether to be rescheduled to a different resource, and the simulation can be set to restart from a few-tasks back. Thus, the simulation framework could support the developer to plan and optimize resources for real scientific simulation

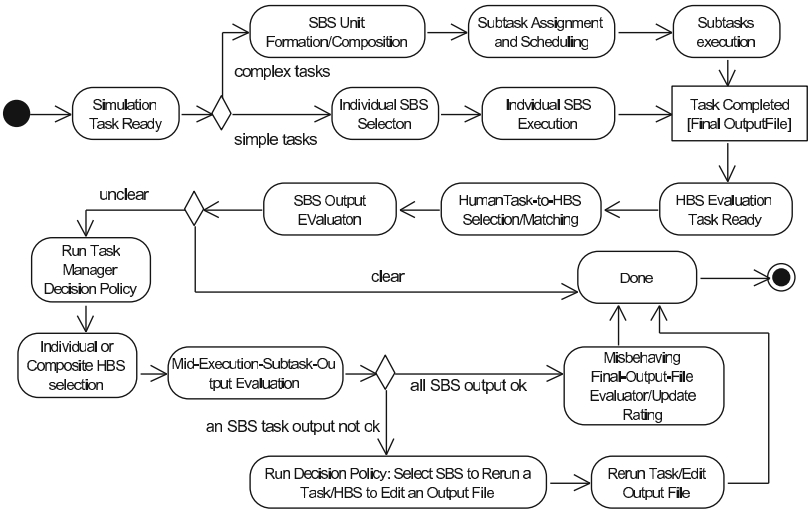

Fig. 3. Human-in-the-loop scenario

workflows. Figure 3 shows a specific case where an HBS is selected to evaluate the final QoR only after all tasks of the simulation are executed. In the case this HBS evaluates a bad result, a notification is sent to the task manager which can run a decision-policy about which task's output within the workflow should be chosen for QoR evaluation. For each of the chosen tasks an appropriate HBS is selected to inspect task results. If a task result within the workflow is evaluated as wrong the task manager is notified,so it can decide whether to reschedule the specific task on the same SBS, select another SBS and assign the task to it or select an HBS to edit the output file. If all the selected HBS for evaluating QoR of intermediary results indicate that task results are correct it means that the initial HBS selected for the final SBS QoR output evaluation is misbehaving and malicious, hence its trustworthiness and rating scores are updated.

Some of the capabilities, specific to this scenario, that our simulation framework design supports, are:

- Creating human tasks, the input data of which is the SBS output;
- Providing decision-making algorithms for injecting HBS to conduct result analysis or editing;
- Selecting the most appropriate resource for QoR evaluation;
- Monitoring and testing with updating HBS profile and performance based information.

More specifically, the framework could support different factors influencing decision-making, e.g., results from conducting tradeoff analysis of cost/QoD versus time, depending on the requirements of the tasks. Social choice theory research results could be integrated into the framework to support the selection

of voting schemes which should be employed when intermediate HBS evaluators are used to evaluate and correct SBS output, to decide on the correctness of a final QoR evaluators assessment (e.g., to simulate the case when a final QoR HBS evaluator has assessed the final result to be incorrect and thus other HBS evaluators are injected after each SBS task when the same workflow is simulated anew).

6 Conclusions and Future Work

In this paper we present a simulation framework to support research on the development of socially-enhanced applications that need both software-based and human-based services. We have described an overview of the conceptual design of our framework discussing the need and use of each component. Thus, we have shown the need to have a hybrid simulation framework and the requirements that the framework needs to fulfill. As we are in the early stages of development, our future work concentrates on further implementation of the simulation framework with a particular focus on both centralized and decentralized mixed resource scheduling algorithms.

References

1. Bermbach, D., Kern, R., Wichmann, P., Rath, S., Zirpins, C.: An extendable toolkit for managing quality of human-based electronic services. In: Human Computation. Volume WS-11-11 of AAAI Workshops. AAAI (2011)
2. Bernstein, M.S., Little, G., Miller, R.C., Hartmann, B., Ackerman, M.S., Karger, D.R., Crowell, D., Panovich, K., Ma, C.: Soylent: A word processor with a crowd inside. In: Proc. UIST 2010 (2010)
3. Buyya, R., Murshed, M.: Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. Concurrency and Computation: Practice and Experience (CCPE) 14(13), 1175–1220 (2002)
4. Calheiros, R.N., Ranjan, R., Rose, C.A.F.D., Buyya, R.: Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services. CoRR abs/0903.2525 (2009)
5. Chen, K.T., Wu, C.C., Chang, Y.C., Lei, C.L.: A crowdsourceable qoe evaluation framework for multimedia content. In: Proceedings of the 17th ACM International Conference on Multimedia, MM 2009, pp. 491–500. ACM, New York (2009)
6. Paolacci, G., Chandler, J., Ipeirotis, P.G.: Running experiments on amazon mechanical turk. Judgment and Decision Making 5(5), 411–419 (2010)
7. Schall, D., Truong, H.L., Dustdar, S.: Unifying human and software services in web-scale collaborations. IEEE Internet Computing 12, 62–68 (2008)
8. Shahaf, D., Horvitz, E.: Generalized task markets for human and machine computation. In: Fox, M., Poole, D. (eds.) AAAI. AAAI Press (2010)
9. Truong, H.L., Dustdar, S., Bhattacharya, K.: Programming hybrid services in the cloud. In: 10th International Conference on Service-oriented Computing (ICSOC 2012), Shanghai, China, Novemebr 12-16 (2012)
10. von Ahn, L.: Human Computation. PhD thesis, School of Computer Science, Carnegie Mellon University (2005)