OPAALS PROJECT

Contract n° IST-034824

WP3: Autopoietic P2P Networks

Del3.5 - EvESim extensions for P2P simulation

Project funded by the European Community under the “Information Society Technology” Programme

Contract Number: IST-034824

Project Acronym: OPAALS

Deliverable N°: D3.5

Due date: August 2008

Delivery Date: January 23, 2009

Short Description: This report summarises in brief the reengineering of the EvESimulator from a standalone agent simulation to a distributed simulation framework. The new EvESimulator (EvESim) is implemented now basing on a service oriented architecture which allows to run agent-based simulations on top of the two P2P systems Soapod and Servent.

Author: SUAS (Christoph Adelberger, Thomas Kurz, Raimund Eder, Thomas J.Heistracher)

Partners contributed: SUAS

Made available to: OPAALS Consortium and European Commission Versioning

Versioning

Version	Date	Name, organization
0.1	22/10/2008	(first submission) Kurz, Adelberger, Heistracher (SUAS)
0.2	04/11/2008	Kurz, Adelberger, Heistracher (SUAS)
1.0	20/11/2008	Kurz, Adelberger, Heistracher (SUAS)
2.0	12/12/2008	Kurz, Adelberger, Heistracher (SUAS)
3.0	23/01/2009	(final submission) Kurz, Adelberger, Heistracher (SUAS)

Quality check

Internal Reviewers: Juanjo Aparicio (TI), Sotiris Moschoyiannis (UniS)

Dependences:

Achievements*	As a pivotal step toward the simulation of biological, ecological and evolutionary approaches in WP3, the EvESim framework - developed in the DBE project - was re-engineered and implemented in a fully distributed manner. The current implementation includes only one biological inspired service, which is a genetic algorithm for abstract service composition. The focus of the work at hand was to prepare the infrastructure in order to implement later more biological inspired algorithms. Also the current implementation of the transactional model is just a first step in order to simulate more complex structures on and related to the OPAALS P2P infrastructure later on.
Work Packages	• WP3 deals with the core architecture of Digital Ecosystems in Task 3.6. The present deliverable aims to contribute to this architecture research. The research on transaction models and dynamic service composition relates to work within EvESim. The implementation of more advanced simulation cases on transactions is one major task for the further work on EvESim. • Task 3.8, Biological, ecological and evolutionary approaches for P2P systems is connected very strongly with Task 10.5, Task 10.10, and Task 10.11 which deal with the visualisation of the simulation cases and cross-domain collaboration as regards social networks.
Partners	The whole consortium. Specifically the partners from the above-mentioned Workpackages and Tasks (IITK, IPTI, LSE, TI, UniKassel, and UniS) could benefit from reading this deliverable and the related online documentation.
Domains	As the present deliverable is a code deliverable, the focus is clearly on computer science and software engineering. Nevertheless, the design and architecture of the new service oriented EvESim is relevant to all other domains as well. One focus of EvESim is clearly on the integration and cooperation of different disciplines by the utilisation of one simulation / emulation framework. Although the focus is on integration, of course not all disciplines and activities as regards OPAALS simulation cases are integrated in EvESim, at least not yet.

Targets	This deliverable mainly targets at researchers in the computing domain in terms of the implementation details and findings regarding differences between P2P implementations in practice. Nonetheless, this deliverable acts as the basis for a much broader audience for implementing and designing cross-domain simulations / emulations with EvESim.
Publications*	R. Eder C. Adelberger, T. Kurz and T. Heistracher. Evaluation of two p2p-approaches for a distributed simulation framework. In 2nd international OPAALS Conference on Digital Ecosystems, 2008.
PhD Students*	Undergraduate : C. Adelberger
Outstanding features*	Specify the outstanding features of the work being done (incremental change in the state of art, improving significantly the state of art, or going beyond) and if anyone outside the OPAALS Consortium has taken notice of this work Next steps are: • Integration of visualisation components • Further improvement of visualisation components • Implementation of import functionality for social network data (starting with topologies) • Creation of a generic agent model which can be easily extended by other stakeholders • Setup of new simulation cases together with UniS, IPTI and other partners from the consortium • Close collaboration with IPTI, T6 ECO and other social science partners for deeper understanding of social variables for setting up social network structures for simulations Outside the project, the developers of Soapod are collaborating and adding features also for the benefit of OPAALS work. EvESim is linked from the http://www.soapod.org/ webpage. Furthermore, the research group around Pietro Terna (see also http://web.econ.unito.it/terna/), who are working on agent-based simulations and SWARM, contacted SUAS as regards an collaboration on the basis of EvESim for an FP7 project, which unfortunately, was not funded. It is planned to go for a new proposal this or next year.
Disciplinary domains of authors*	R. Eder (Information Technologies, Software and Systems Engineering) C. Adelberger (Information Technologies, Software and Systems Engineering) T. Kurz (Information Technologies, Software and Systems Engineering, Interpersonal Communication) T. Heistracher (Information Technologies, Software and Systems Engineering, Biophysical Modelling)

The information marked with an asterisk (*) is provided in order to address Recommendation n. 4 from the Year 2 review report

Contents

Executive summary

As a pivotal step toward the simulation of biological, ecological and evolutionary approaches in WP3, the EvESim framework – developed in the DBE project – was re-engineered and implemented in a fully distributed manner. This distributed simulation toolkit is the basis for the following major tasks of EvESim within OPAALS:

- Social network analysis (WP10)
- Transaction execution and simulation (WP3)
- Testing infrastructure by simulating on P2P (WP5)
- Network visualisation in simulation and real-world networks (WP10)

The whole up-to-date documentation of the distributed EvESim is available at `http://www.evesim.org` including also Javadoc documentation with the codebase. The sourcecode of the P2P release of EvESim can be found in the *evesim2* folder at `https://evesim.svn.sourceforge.net`.

Because this work is a code deliverable and because the documentation is available online in a very detailed form that includs installation guides and results of the first simulations, this document acts just as the official document to report the dependencies and project-specific tasks for EvESim.

1 Introduction

The *EvESimulator* (EvESim) was intended to be a simulation framework for biologically inspired P2P systems – the so called *Evolutionary Environment* (EvE) as a part of a Digital Ecosystem. It focusses clearly on providing a framework for various simulations depending on a number of parameters and not a distinct simulation itself. Although its focus is on the EvE, the EvESimulator simulates a DBE or – depending on the use case – Business-, Computing- or Social-Networks. The difficulties, the EvESimulator faced in the case of simulating the EvE are outlined in the following chapter. Thus the EvESimulator can be comprehended as an collaborative platform for interdisciplinary research acting as a framework for understanding, visualising and presenting the DBE concepts to contributors.

Though, the name *Evolutionary Environment* and the name of the simulation framework, *EvESimulator*, suggests to have evolutionary and biological inspired algorithms implemented, the current implementation includes only one biological inspired service, which is a genetic algorithm for abstract service composition. The focus of the work at hand was to prepare the infrastructure in order to implement later more biological inspired algorithms.

In order to analyse non-linear and adaptive interactions, computer simulations are used. The basic idea of such computer-aided simulations is to specify the rules of behaviour for individual entities as well as their interaction. The simulation itself aims at simulating a multitude of individual entities using a computer model, and to explore the consequences of the given rules. As the individual entities are usually implemented in form of agents, the simulation of their behaviour and interactions is known as an agent-based simulation.

The particular innovation in the EvESim is the interdisciplinarity of contributors as well as the extendability. For example, findings from social science regarding the critical variables for SME behaviour are implemented in the simulation to provide a more realistic interaction behaviour. These settings can then, for example,

be included in a computational simulation of transactions in any kind of P2P network implemented and coupled with EvESim. As the simulation framework is very generic and open for extensions, it offers also many options for cross-domain simulations. Additionally, these can exemplarily show differences of network behaviour for external stakeholders in an comprehensible way.

2 EvESim Definition and Goals

Additional to the general introduction to evesim the following chapter outlines the various strengths of activities related to EvESim. The structure is similar to the presentation of EvESim at the OPAALS review, 2008 in Brussels. The following subsections should give a more detailed insight into the dependencies of EvESim additional to the sourcecode. Although this is a code deliverable, we include this chapter for 1) clarifying the subject and dependencies, 2) sketch parts of the implementations and reasoning which will be reportet in *D3.7 Biological and Evolutionary approaches for P2P systems and their impact on SMEs* and *D10.8 Report on Social Networks in cross-domain collaboration* as well as 3) answer the detailed questions of the internal reviewers.

Biological inspiration

Since the beginning of the Digital Business Ecosystem (DBE) research the term DBE was used for referring to several paradigms and concepts - some implemented and used by SME communities, some not. If we are speaking of Digital Business Ecosystems we are referring to the paradigm which is summarised in [1], referring to natural science paradigms especially in [2]. Nevertheless, it has to be stated that the term evolves as well over time, which results with emphasis, e.g. on the digital aspects of DBEs like in [3] and related to SOA in [4].

The biological inspired concepts out of this DBE project manifested themselves in a infrastructure component which was called the Evolutionary Environment (EvE). With the implementation of the EvE within the core architecture of the DBE (see [5]) it was intended to spread the most useful and requested services over the DBE network and therefore have a kind of biologically inspired promotion mechanism. For this reason, genetic algorithms together with an appropriate fitness function were identified as the best solution for enforcing these biologically inspired mechanisms. Although, the infrastructure component of the EvE was implemented and integrated,

it was not possible to find one single fitness function which fits to all kind of services in a DBE [5]. Additionally, 1) the number of services available at the end of the DBE was far too small for a useful test of different configuration parameters, and 2) it was impossible to test a single fitness function for the heterogeneous types of services at this stage.

Though service composition on the basis of agent chains (see [6] and [7]) as well as simulations on the basis of DNA strengths or attribute-value pairs (see [8] and [9]) worked considerably well for simulating the EvE, the implementation of a fitness function for DBE services which are described with a Business Modeling Language (BML) (see [10]), Semantic Service Language (SSL) (see [11]), as well as a Service Description Language (SDL) (see [12]) turned out to be difficult, especially with no real services for tests.

As the simulations with EvESim in the DBE project were successful and promising; the re-usage of EvESim seemed to be fruitful, which was also acknowledged by the reviewers during the last review. The shift in focus from software services toward more generic knowledge services or knowledge resources make also the search and composition of services easier, because they don't have to fit 100 % in terms of technical interfaces descriptions. By the use of Natural Language (NL) and the structured approach of SBVR, the complexity of service descriptions increased, but we have a much better perspective for a critical number of services. Large social networks like the ones in india (DEAL project) and Brasil (guigoh community) make the application and test of the developed paradigms possible.

Nevertheless, some modifications in the code base of EvESim and a service oriented approach even for the simulation framework was needed to adapt to the needs of OPAALS. Parts of these modifications were documented in D3.3 and further will be reported in D3.7, D10.8 and others (see also [13]). In order to enable these modifications, the work on the evolutionary parts was postponed. Even so, we kept the proof-of-concept simulation cases for critical mass and clustering (see [9], [14]). In order to evaluate, if biological inspired algorithms can improve existing standards, e.g. in P2P for node and service lookup, first a simulation framework has to be created which is capable of simulating and testing these algorithms. It is intended to take over some of the outcomes of WP1 on Phase III and include them into EvESim.

At this point it is essential to understand what EvESim is and what it is not. EvESim is a simulation framework and since the last release a testing tool, which enables cross-domain collaboration on a common code base. Here an example: In

WP 10, IPTI is working on the analysis of social network behavior and the social variables which are underpinning the structure (topology) and behavior (request for services/resources) of a network. This research influences the EvESim in setting up close-to-reality networks for simulations. In WP3, UniS is working on transactions. As soon as the basic infrastructure for transactions is implemented, these can be simulated based on the network topology, coming from WP10. Furthermore, the simulation is not only a simulation at this stage, because the simulation itself will run on the digital ecosystem infrastructure. That enables not only to simulate response times of service, but call these services and test them on a large network, before OPAALS has such large networks. The results of these simulations and tests can finally be used to adjust and recursively improve the infrastructure and the simulated algorithms. Everything on the real infrastructure and before users could probably get upset by the first usage of an alpha release.

Before the following sections go into detail about the fields of application for EvESim, in the following a few clarifications on the current implementation and the design decisions of components, implemented in the DBE and OPAALS version of EvESim.

When we use the term *service composition*, we mean a set of services, as a answer to a certain request. *Abstract service composition* refers to the composition of abstracted services in a simulation for example. The difficulties in finding services in terms of technical interfaces were discussed and documented exhaustively already in the DBE project. For simulating service requests and for the search by using, e.g. genetic algorithms, we refer to the abstraction of services which we have taken over from the DBE project (see [6], [7], [15], [16], [8], [9], [14]).

The genetic algorithm (GA), which is used in EvESim was a joint development of University of Birmingham (responsible for optimisation research in the DBE project) and SUAS. As stated in [9] and [17], we implemented the genetic algorithm in the EvESim framework as an exchangeable module from the beginning on. Aligned with the generic and service oriented architecture of EvESim, the genetic algorithm should be interchangeable, so that other researches with a focus on optimisation research can test new algorithms for the applicability for digital ecosystems by implementing a GA service and plug it into EvESim. As can be seen from [9], we tested in the first version of EvESim different GAs and also an implementation of the so called UMDA algorithm and even made the implementations exchangeable for the user. Because of the limited resources for the refactoring of EvESim, we migrated only

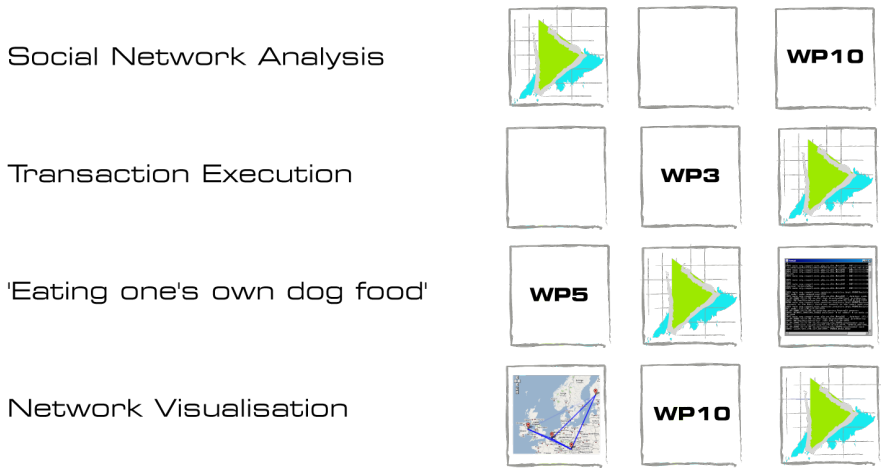

Figure 2.1: Structure of EvESim related work and strengths of activity.

the self developed algorithm to the distributed EvESim version.

Structure of Activities

In the following sections the current strengths of activities as regards EvESim are described. The structure, outlined in Figure 3.1, was presented at the OPAALS review 2008. Additionally to the core functionalities according to the Description of Work (DoW) we try to find new collaborations with partners along the development of EvESim. This is aligned with the comment of the reviewers that *[...] the use of EveSim may be crucial in Phase 3 in order to find results by simulation [...]* and *[...] Since it is mature, it should be more aggressively used by other parts of OPAALS. [...].* It has to be pointed out here that depending on a new prioritisation of tasks, the introduction of biological inspired computing and the efforts in transaction simulations could suffer from the additional interfaces to partners and additional tasks, given the low resources on this particular task.

2.1 Social Network Analysis

Per definition the EvESim is a simulation framework used for cross-domain collaboration, simulation and testing. The user interface of EvESim allows the user to configure various settings in a Graphical User Interface (GUI). As the EvESim GUI

is currently customised to the OPAALS needs, we briefly show here the old settings windows in order to get a feeling for the type of user interaction and configuration we are including also for OPAALS. Figure 2.2 shows the EVESim Settings window which acts as the main configuration window for the EvESim. It enables the user to configure the main characteristics of an simulation without changing the source code. That is particularly important for non-technically experienced groups with a broad knowledge in networks or social interaction.

The EvESim Settings is structured in four configuration tabs, namely (1) General Settings, (2) GA, (3) Comparisons and (4) Equilibrium. Moreover, at the bottom of the window the import and export features are situated. Details on the EvESim configuration in DBE can be found in [9]. For OPAALS we intend to have similar settings, customised to the OPAALS needs and dependent on the upcoming simulation requests. The graphical representation of the network simulation will be based on google maps. The prototype for that visualisation was already documented in *D10.7 Visualisation Service for P2P infrastructure and EvESim based on GoogleMaps*. The same visualisation can also be used for the topology visualisation of the real P2P network. The XML data structure for this visualisation is done together with Tampere University of Technologies.

The next level of settings are the configuration of the network topology and behavior of a social network by applying social variables. The origin of this effort was in a collaboration with Antonella Passani on identifying social variables which identify SMEs in a social network. The first efforts were described in *D10.7* and finally also in [18]. Although the first implementation was only using a few variables for setting up the topology of the network, current activities in WP 10 are working further on customising the so called "'prototypes"' of SME behavior along variables like attitude, ICT usage, social capital etc..

Additionally, the behavior of the actors in a simulation should be influenced by the outcome of social science. Therefore, in *T10.11 Social networks in cross-domain collaboration* and in *3.8 Biological, ecological and evolutionary approaches for P2P systems*, especially IPTI is working on the social variables and on the behavior of actors in social networks (see also [19]). EvESim will not simulate all related networks, but builds the platform where the results of the social research of IPTI can be implemented for the benefit of other simulations. One example could be that the request behavior of OPAALS users is studied and the frequency of using a certain service can be taken over for simulating the transactions underneath these

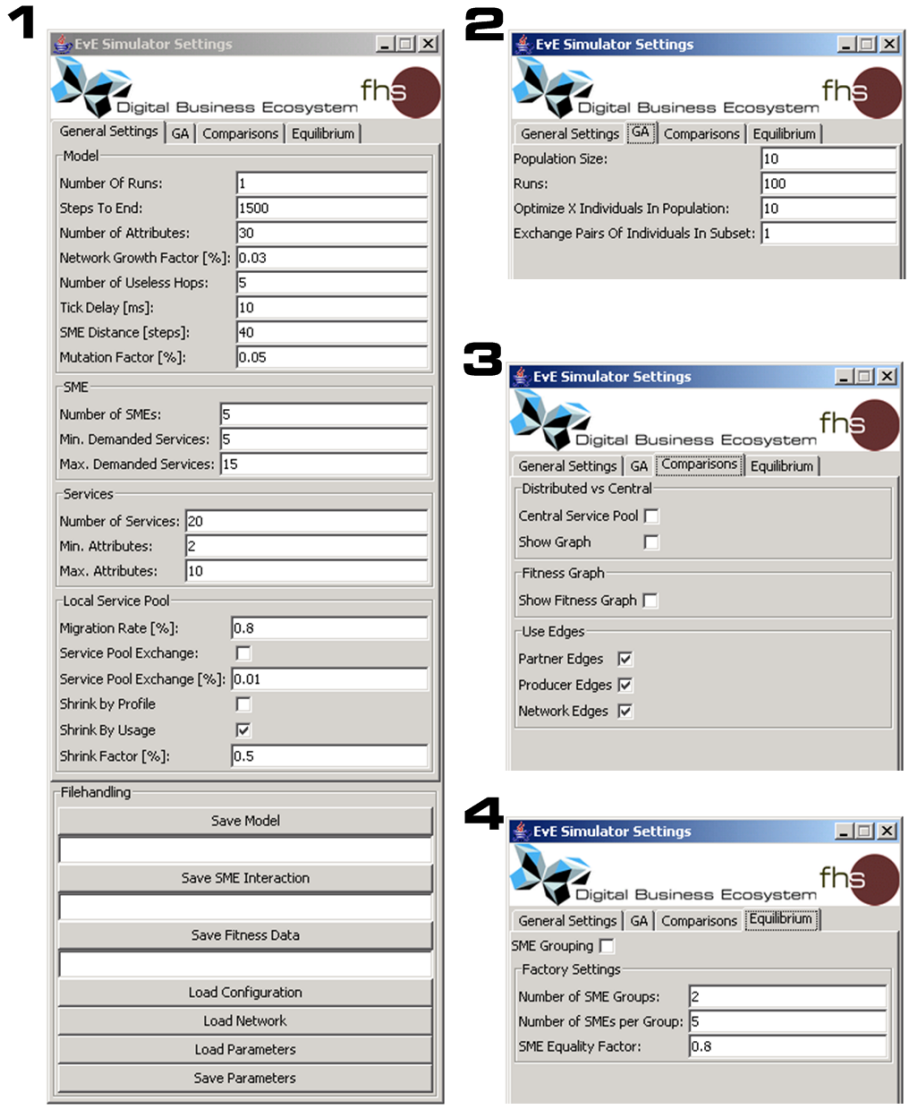

Figure 2.2: Customized configuration window of the EvESimulator in DBE. [9]

service requests or usage.

2.2 Transaction Execution

As EvESim is part of WP3 it is also connected to the simulation and test of the transactional model, reported in [20], [21] and [22]. The first results of integrating the transaction model in evesim were reported in [22]. As the purpose of EvESim primarily focuses on simulations in distributed / P2P environments this is a coherent step in order to find out more about the structure of the P2P network that is formed through the local interactions (business and knowledge services) which follow the basic principles of the transaction model [21].

Starting by a prototypical simulation of transactions (see also [22]) we want to emulate the behavior of transactions now in a real-world environment. The simulation of transactions build the basis for moving on to Virtual Private Transaction Networks (VPTNs), Distributed Virtual Super Peers (DVSPs) and finally the next generation of the OPAALS P2P network. Based on the research, described in [20], [21] and [22], SUAS will emulate the computational, real-world, runtime characteristic of the upcoming OPAALS P2P infrastructure while it is designed and implemented. All this, is work in progress but helps in direct feedback to the implementation group and a test loop parallel to the implementation of the P2P network infrastructure.

As described in [22], we started by creating a prototypical implementation of the Local Coordinator component with a minor subset of functionality in order to generate a working prototypical simulation. Because EvESim is doing real service calls, the concept of Coordination had to be implemented. That was done for a prototypical and small use case, but needs to be extended considerably when dealing with more complex functionality.

In order to point out what is needed to do transaction simulations in future we are giving here more information what will be needed for whatever implementation group for implementing the Coordinators in real-world. The term "'simulation"' in the context of transactions and also in terms of the infrastructure which will be described in the next subsection is a bit missleading. As EvESim works on a P2P infrastructure and is not a local simulation toolkit, the simulations of transactions are more an "'emulation"' of the behavior. That means that when we are simulating a transaction, real services on the nodes are called. For this simulation / emulation

we are using at the moment simple wait-services and observe runtime behavior and failure rates as described in [22].

In the following, the defined functionality of the Local Coordinator is listed, reffering to [20], Section 2.3.2.1, and [21], p31. The parts partially implemented are in *italics*.

1. coordination of the transactional
    - data oriented coordinators
    - *sequential process-oriented coordinator*
    - *parallel process-oriented coordinator*
    - *sequential alternative coordinator*
    - *parallel alternative coordinator*
    - delegation coordinator
2. service orchestration
3. keeping logs for managing dependencies
4. implementing a recovery mechanism
5. handling the low bandwidth
6. dealing with the low processing power

According to the model, the Coordinator is part of an agent which has additional components. The interfaces between the components are introduced in [21]. Sequence diagrams are provided as well, although it would be helpful to have the individual coordinators included in the sequence diagram as well in order to visualize the message flow between the components involved in a transaction. To implement the Local Coordinator the interface descriptions need to be more detailed: e.g. the messages between the Agent and the Local Coordinators and the messages that are sent between the involved Local Coordinators are not provided yet.

For the purpose of the simulation a interface was assumed: the LocalCoordinator (as well as Web Services and Service Compositions) all got an execute method. As in- and out-Parameter an Transaction Pipeline object is expected / returned. This latter approach was borrowed from industry integration products. Although this is

working for our simulations this approach is taken from a conventional Hub-and-Spoke architecture and probably not valid for a release of the OKS.

The following source code shows the interface between the Coordinators and Services:

```
public interface TXService {

public TxPipeline execute(TxPipeline p) throws TxExceptionImpl ;

}
```

The prototypical Local Coordinator is able to invoke services sequentially and in parallel. Processing of data can be added when the required specification is available for that. This is also true for all other features mentioned in D3.3.

Note that the failure mechanism was implemented using Exceptions, which will not be valid in a real world implementation of the transactional model as it only supports the commit / rollback scenarios. Scenarios like forward recovery will be harder to implement using this way. Scenarios using omitted results probably are impossible using this approach. Details that will be addressed in future deliverables, clarifying these open points about the transactional model implementation, were assumed in order to move on with the simulation / emulation of the transactional model.

It has to be stated here as well, that for a implementation of the transactional model (see [20] and [21]) an full implementation of the Coordinators and the described functionality is needed anyhow. The benefit of using EvESim is clearly on the simulation / emulation of the behavior before it will be released to the public. Furthermore, the simulation / emulation has the benefit of testing the system on a large number of nodes and with a high number of service calls without a required user action. Nevertheless, the implementation of the coordinators need to be finished before it can be integrated and tested by SUAS in EvESim. In parallel SUAS is working with IPTI on the integration of the new Jxta P2P implementation with EvESim, which is also necessary to simulate / emulate the OPAALS P2P platform of choice.

2.3 Using and Testing the Infrastructure

Before going into detail on the benefits and goals of using and testing the OPAALS infrastructure we want to define in the following why the current EvESim is running on Soapod and Servent and not on another infrastructure.

As we wanted to use the infrastructure, which was developed in DBE and OPAALS for EvESim as well, the decision for using that application container was clear (we used the most recent Version at that point in time, which was Servent V0.3.14 and later V0.3.15). On the contrary, already in DBE and also in OPAALS, some partners felt uncomfortable with using the flooding approach of the Servent by running FADA technology. Additionally, the Servent was an in-house development and the choice for implementing a service container, whereas there are open source application servers based on Tomcat, for example, which are providing similar functionality, was often criticised. Out of these and other reasons the Soapod project was created by a group of developers from DBE in their free time. Although they promised a better compliance with standards and another node-lookup strategy, it was never tested in direct comparison to the Servent.

As the cost for implementing the EvESim on a second platform was low, we decided to that and compare Servent and Soapod on the basis of criteria like Implementation, Configuration, Build, Service Lookup etc.. More detailed information on the differences can be seen at `http://evesim.org`. In summer 2008, IPTI came up with the idea of using JXTA (see also `https://jxta.dev.java.net/`) for the OPAALS infrastructure. Similar to Soapod, we hope to make a considerable input for the open source community by comparing Jxta similar to Servent and Soapod on the EvESim webpage.

The extensions on EvESim have to be seen as an ongoing process, trying to provide a simulation and testing framework also for the latest design decisions. The simulations on transactions, documented in [22] were done on a first alpha release of the distributed EvESim. As this document is a add-on to the code deliverable of EvESim, parts of this code was therefore implemented for the simulations for [22] already.

The refactoring of EvESim and moving it to a service oriented architecture results in a generic usage of EvESim for many P2P infrastructures. As the modules of EvESim are encapsulated functionalities with clear interfaces, only the interfaces, node lookup, service lookup and service calls have to be changed for moving to another platform. One example of this benefit is the migration from Soapod and

Servent to Jxta. Consequently, EvESim can be migrated to future P2P systems or service oriented architectures of choice, with minimal implementation efforts. As there is no in depth expertise on Jxta in OPAALS yet, we are currently collaborating with IPTI and exchanging experience on the implementation issues of Jxta.

Additionally, here a short note on the general problem of flooding and the Servent at the moment. Flooding algorithms have been proven not to scale (see [23]). Consequently, if one was to search for a particular service in a infinitesimal sized P2P network using a flooding algorithm the query would either run and block forever or simply timeout. If then the depth of search is limited, you cannot search the whole network and so it is never sure if there is a solution to the search query. The plain fact of the matter is that FADA uses a naive scheme for searching and its flooding techniques are basic. At a technical level FADA is also difficult work as it is prone to OutOfMemoryExceptions and as such is not reliable (that was experienced a number of times at a code camp in Zaragoza in '05 for example and was one of the reasons for refactoring the Servent). The Servent itself while runs somewhat reliably via Jetty does not provide services using any standardised technologies and whereas integration is possible adopting non-standard methods only add to the Servent adoption barrier. On the communications of the Servent - the Servent is dependent on Java proxies. This in turn depends on Java serialisation. For this to work, all FADA nodes with service proxies must have the same version of serialised proxies - otherwise those with a previous serialisation version cannot be understood by those of the latest version. This is a problem not of Java but of new versions of certain Servent libraries.

On the contrary, Soapod is using DHTs for the distributed index of nodes. Details can be found on `http://www.soapod.org` and in the source code documentation. The scalability of DHTs can be found extensively in literature. Although there were preliminary observations on complexity and scalability in [20], there is still no implementation of a full scalable P2P network in OPAALS at this point in time. Therefore Soapod (with DHTs) will be supported for the moment and in parallel we are working on an integration of Jxta, where the scale-free approaches, described in [20] will hopefully be implemented so that we can test them in EvESim as well.

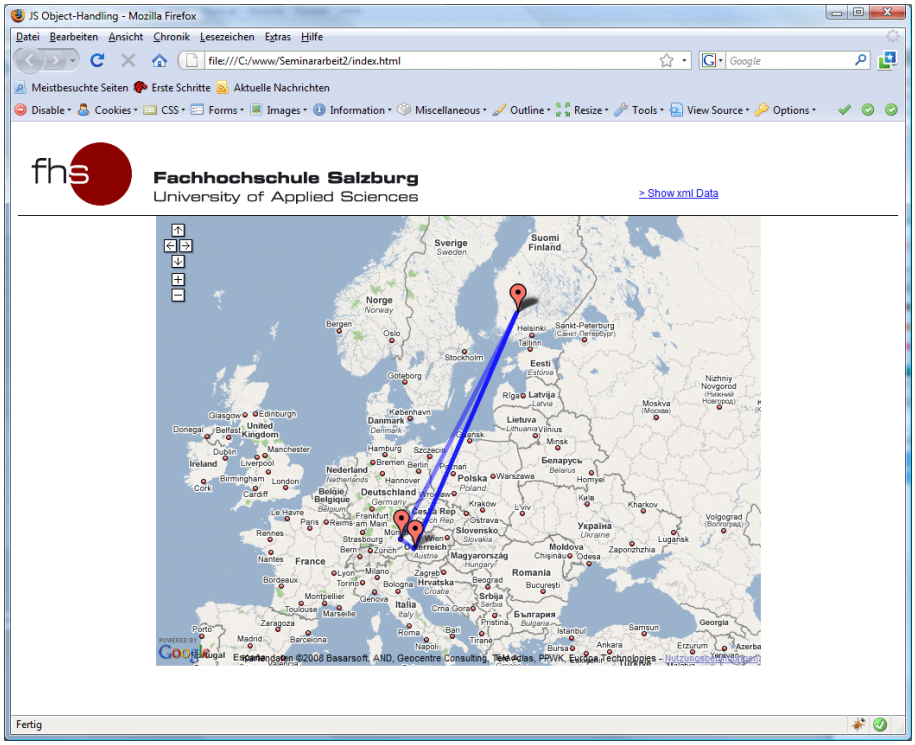

Figure 2.3: Example of a network topology visualisation on a Google Map. [24]

2.4 Network Visualisation

The visualisation components of EvESim were developed until now in strong collaboration with TUT in WP10. [24] describes in detail which components are used and how Ajax was used to visualise the network topology on a Google Map. Figure 2.3 shows a screen-shot of such a network visualisation. The visualisation generic, which means it can be used for the visualisation of an simulated network in EvESim as well as for the visualisation of the infrastructural network nodes. At the moment it supports only Soapod but will be migrated to Jxta as well. The basis of the visualisation is an XML file which was developed with Tampere University of Technologies (TUT) and in synchronisation with IPTI. That was necessary in order to enable future import and transformation capabilities from other social network tools. More detailed information of the Visualisation can be found in [24].

3 Distributed EvESim Architecture

In order to overcome the limitations of the stand-alone version of the EvESim and to provide an adapted simulation framework, the EvESim was redesigned and encapsulated into several basic entities. These entities were distributed on a P2P network, which is built on the same concepts as the Digital Business Ecosystem (DBE). The current implementation of EvESim supports the so called Servent and the Soapod network. These P2P implementations will be described later on in this deliverable. The intention of implementing EvESim ob both systems, was to have a means of comparison and testing for P2P systems which are completely different in their implementation, but offer the same basic functionality. With some adaptations other P2P implementations can be used and tested as well. The distributed simulation framework architecture opens the possibility to share simulation resources and to simulate network traffic and network connections in a more realistic manner. The EvESim consists of several basic functional units, such as a *simulation control unit*, an *agent container unit*, a biologically inspired *optimisation unit* and a *reporting unit*. Consequently, it seemed obvious to extract those units and encapsulate them into services and form a service-oriented framework. These services are:

- CoreSim
- Agent Pool
- Agent
- Genetic Algorithm
- Status Report
- Central Service Pool
- Maps

- GUI

The following subsections describe these logical entities in more detail. This document just provides a brief overview of the EvESim architecture. Detailed descriptions including installation guides and HOWTOs can be found at `http://www.evesim.org`.

CoreSim

Representing the first service, the *CoreSim* is intended to be the only centralised service in the simulation framework. It represents the single entry point of a simulation and provides configuration and evaluation capabilities via a Graphical User Interface (GUI). This GUI is used to configure the behaviour of the agents on the one hand, and to control the simulation itself on the other hand. Furthermore, the CoreSim is responsible for initialising the Agents (see below), such as creating services and setting up parameters based on the settings given by the user. After deploying the agents in the Agent Pools (see below), the CoreSim collects the simulation data and proceeds with evaluation and serialisation tasks.

Agent Pool

The second service is defined as the Agent Pool. The Agent Pool represents a container of Agents in the simulation. Depending on the configuration an agent pool can have several agents and a simulation can have several agent pools. The Agent Pool defines the environment for the agents and provides interfaces for communication and interaction with other Agents and the DBE, respectively.

Agent

The Agent defines and manages the activities of a Small and Medium-sized Enterprise (SME) and represents the main actor in the simulation. Although, each agent could have been implemented as a service as well, it was decided to aggregate several Agents in an Agent Pool in order to reduce the number of communicating services in the simulation network.

Genetic Algorithm

The third service is named *Genetic Algorithm* (GA) and implements the biologically inspired optimisation component of the simulation framework. Based on a common interface, the network can provide several different implementations of GAs. The GA service is called directly by an agent when it gets the authorisation to request a service from the DBE. GAs are search algorithms, which are based on the mechanics of natural selection and natural genetics. In general, GAs can be split up into four operations: (i) selection, (ii) reproduction, (iii) crossover, and (iv) mutation.

With regard to the EvESim, the GA service can be explained by a very simplified example: Considering a hotel room service, the example is extended by a hotel which wants to offer a holidays weekend service. This service offers a hotel room and a candle light dinner on Saturday night. Obviously, the hotel can only offer the hotel room and hence it has to define a service on demand which states the need for a candle light dinner. A restaurant offers a candle light dinner service. Within the DBE, the GA is used to produce the combination of the hotel room service and the candle light dinner service. This service combination is returned to the hotel which wants to offer a holidays weekend service. Consequently, it offers a service combination of separately available services on the network.

Status Report

The fourth service is the Status Report, which represents the communication channel between the agents in the network and the CoreSim service. As outlined before, the CoreSim provides functionality to evaluate and present simulation data and therefore it is dependent on a service which provides up-to-date simulation data. However, each agent manages its own data and is responsible for delivering the data to the status report. This collected simulation data can be retrieved and analysed by the CoreSim. Consequently, the status report acts as a push-and-pull queue for storing information.

Central Service Pool

An additional service represents the Central Service Pool. This service pool provides a central service container for all available services on offer in the P2P network, or the DBE respectively. As the DBE is highly decentralised, this service is not part of the

DBE architecture but was necessary for the simulation framework in order to analyse the behaviour of the decentralised service reproduction and service distribution.

Maps

The visualisation of the whole network of agents and the underlying infrastructure is implemented in the service named Maps. This service provides visualisation components based on Google Maps and Asynchronous JavaScript and XML (Ajax). It is capable of visualising different networks, which evolve during a simulation, such as social networks, producer-consumer networks etc., and provides a user interface to retrieve up-to-date simulation data. These networks are described in an XML tree and are read by an AJAX application and visualised by Google Maps. The XML description can also be used for serialisation or other visualisation of the network.

Graphical User Interface

The *Graphical User Interface* (GUI) is attached to the CoreSim and represents the configuration and visualisation entity of the EvESim. It is used to configure and control simulation scenarios. All these services, also called core services of the EvESim, are distributed on the P2P infrastructure and consequently form a distributed and scalable simulation framework without centralized authorities.

The P2P infrastructure is built on top of service-based P2P network nodes, which act as a network of dynamically connected application servers. These application servers are suited for the use within a DBE and rely on the concept of such DBEs. Consequently, they provide an environment to offer and consume services. Figure 3.1 shows the set up of the distributed EvESim. This figure disregards one component, namely the Graphical User Interface (GUI), as it is attached to the CoreSim. Within this environment, the EvESim distributes its resources and simulates use case scenarios on top of a DBE-like infrastructure. This enriches the results of the simulation framework.

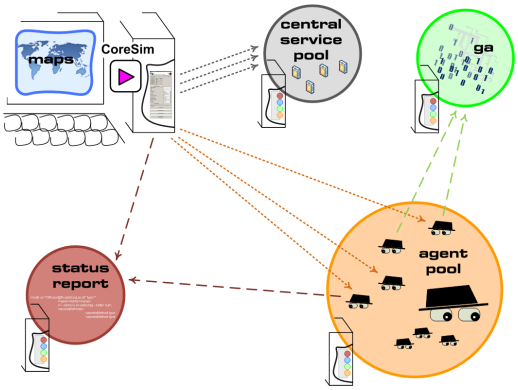

Figure 3.1: Representation of an Actor (SME or agent, respectively) within the EvESim.

4 Conclusion and Outlook

The current version of the EvESim will be extended in WP3 and WP10 mainly. Beside the integration of the visualisation components coming from Task 10.10, the emulation of WP3 outcomes, e.g. transactions, VPTNs and DVSPs, will help in testing the real-world behavior of WP3 outcomes and underpin the relevancy to practice. The start for that will be the utilisation of local coordinators and the migration of the EvESim services to Jxta. Sumarising, the next steps are the following:

- Integration of visualisation components
- Further improvement of visualisation components
- Implementation of import functionality for social network data (starting with topologies)
- Creation of a generic agent model which can be easily extended by other stakeholders
- Setup of new simulation cases together with UniS, IPTI and other partners from the consortium
- Close collaboration with IPTI, T6 ECO and other social science partners for deeper understanding of social variables for setting up social network structures for simulations

Beside these activities we plan to integrate outcomes of WP1 and learn from biological computing for enhancing the current P2P based simulations. Nevertheless, the clear focus is to provide and adapt the simulation framework for the convenience of other partners and for the interdisciplinary understanding of P2P-based social networks.

Bibliography

[1] Edited by: F. Nachira, A. Nicolai, P. Dini, M. Le Louarn, and L. R. Leon. *Digital Business Ecosystem.* European Commission, Directorate General Information Society and Media, 2007.

[2] G. Briscoe and S. Sadedin. Natural science paradigms. In P. Dini M. Le Louarn F. Nachira, A. Nicolai and L. Rivera León, editors, *Digital Business Ecosystems.* European Commission, 2007.

[3] S. Sadedin G. Briscoe and G. Paperin. Biology of applied digital ecosystem. In *IEEE First International Conference on Digital Ecosystems and Technologies*, 2007.

[4] S. Sadedin G. Briscoe and G. Paperin. Digital ecosystems: Evolving service-oriented architectures. In *IEEE First International Conference on Bio Inspired mOdels of NETwork, Information and Computing Systems*, 2006.

[5] Soluta.Net. D21.4 - Detailed Technical Architectural Models. Digital Business Ecosystems Project, April 2005.

[6] G. Briscoe. D6.1 - Self-organisation in Multi-Agent Systems. Version 1, July 2004.

[7] Gerard Briscoe. D6.2 - Control of Self-Organisation and a Performance Measure. Digital Business Ecosystems Project, April 2005.

[8] T. Heistracher, T. Kurz, G. Marcon, and C. Masuch. D9.1 - Report on Fitness Landscape. Digital Business Ecosystems Project, April 2005.

[9] H. Okada T.J. Heistracher T. Kurz, G. Marcon and A. Passani. D9.2 - Report on Evolutionary and Distributed Fitness Environment. Digital Business Ecosystems Project, June 2006.

[10] V. Cisternino E. Caputo A. Corallo, M. De Tommasi. D15.6 - BML Framework final release. Digital Business Ecosystems Project, 2006.

[11] U. Pernice I. Dumitrascu, G. Montanari. D16.4 - SSL Compiler. Digital Business Ecosystems Project, 2006.

[12] Soluta.Net. D16.4 - SDL specification for the DBE Platform Part 1 - SDL Full Definition. Digital Business Ecosystems Project, 2006.

[13] R. Eder C. Adelberger, T. Kurz and T. Heistracher. Evaluation of two p2p-approaches for a distributed simulation framework. In *2nd international OPAALS Conference on Digital Ecosystems*, 2008.

[14] R. Eder T. Kurz, J. Noguera and T. Heistracher. Peripheral simulation framework for digital ecosystems. In *1st international OPAALS Conference on Digital Ecosystems*, 2007.

[15] G. Briscoe. Digital ecosystems: Evolving service-oriented architectures. In *IEEE First International Conference on Bio Inspired mOdels of NETwork, Information and Computing Systems*, 2006.

[16] Gerard Briscoe. *Digital Ecosystems*. PhD thesis, Imperial College London, Department of Electrical and Electronic Engineering, 2009.

[17] T.J. Heistracher G. Marcon, H. Okada and T. Kurz. D16.3 - Report on Adaptive Service Generator. Digital Business Ecosystems Project, April 2005.

[18] A. Passani T. Kurz and T.J. Heistracher. Social network simulation and self-organisation. In P. Dini M. Le Louarn F. Nachira, A. Nicolai and L. Rivera León, editors, *Digital Business Ecosystems*. European Commission, 2007.

[19] F.A.B. Colugnati. Dynamic social network modeling and perspectives in opaals frameworks. In *2nd international OPAALS Conference on Digital Ecosystems*, 2008.

[20] UniS. D3.1 - Preliminary architecture for P2P network focusing on hierarchical virtual super-peers, birth and growth models. Digital Business Ecosystems Project, June 2007.

[21] UniS. D3.2 - Report on formal analysis of autopoietic P2P network, together with predictions of performance. Digital Business Ecosystems Project, August 2007.

[22] CreateNet TechIdeas UniS, SUAS. D3.3 - Full Architecture Definition. Digital Business Ecosystems Project, July 2008.

[23] L. Breslau N. Lanham Y. Chawathe, S. Ratnasamy and S. Shenker. Making gnutella-like p2p systems scalable, 2008. http://www-users.itlabs.umn.edu/classes/Fall-2008/csci8101/papers/gnutella.pdf.

[24] R. Eder T. Kurz, T.J. Heistracher. D10.7 - Visualisation Service for P2P infrastructure and EvESim based on GoogleMaps. Digital Business Ecosystems Project, November 2008.