Using social network technology to provide e-administration services as collaborative tasks

Ourania Hatzi, Mara Nikolaidou, Panagiotis Katsivelis-Perakis, Valentino Hudhra, Dimosthenis Anagnostopoulos

Department of Informatics and Telematics, Harokopio University of Athens, Greece

{raniah,mara,pkatsiv,it20739,dimosthe}@hua.gr

Abstract. This paper presents an approach employing social network technology to facilitate e-administration within collaborative communities. E-administration services are provided through task coordination between specific participants. Task activities are performed by gadgets acting on behalf of participants. Task assignment is based on participant roles and relations in the community, explicitly defined within the social network. Interaction between gadgets is governed by rules based on participant roles, dictating the obligations and responsibilities of each party. Whenever the execution of a certain gadget depends on the previous execution of a series of other gadgets, a recommendation mechanism employing AI planning is used to provide a plan according to which gadgets should be combined. The implementation of a social network platform supporting e-administration based on extending the OpenSocial API is also presented. The proposed platform has been utilized to develop a social network for the academic community, featuring pilot implementation of specific e-administration services.

Keywords: Social Network, Collaborative Tasks, E-administration, Recommendation Mechanism, AI Planning, OpenSocial.

1 Introduction

Corporations and organizations have recently incorporated novel interaction models for knowledge dissemination, intra-organization communication and collaboration between their members, in order to facilitate electronic service provision, leading to e-administration. At the same time, social networks have emerged as a particularly widespread interaction paradigm.

In this paper we explore an approach that employs the familiar social networking interaction model to provide complex e-administration services in a collaborative community. Social network participants collaborate to perform complex e-administration tasks, having in mind their role in the community and the relations between them. E-service provision through task coordination between specific participants is performed by gadgets, corresponding to specific task activities, acting on

their behalf. Interaction between gadgets is governed by rules based on the participant roles, which dictate the obligations and responsibilities of each party. In the proposed service delivery model, complex e-service functionality is not predefined, as for example using BPEL-like languages; instead, gadget compositions resulting to e-service functionality are generated ad-hoc by the collaborating participants. To achieve this, participants should decide the conditions enabling the execution of a certain gadget. In the case that gadget execution depends on the previous execution of a series of other gadgets, a recommendation mechanism may be invoked to assist the participant in gadget composition. Such a recommendation mechanism employing AI planning techniques to provide plans according to which gadgets should interact to complete complex tasks is also discussed.

Implementation requirements to integrate collaborative e-gov service delivery in a social network platform and corresponding proposed extensions of the OpenSocial API, proposed by Google to implement interoperable social networks, has been presented (Dais, Nikolaidou & Anagnostopoulos, 2011). Based on these guidelines, the delivery of e-administration services in an academic social network was implemented, based on the proposed OpenSocial API extensions. A pilot implementation of specific e-administration services is currently available.

The proposed research does not simply targets a custom implementation of an ad-hoc social network for the academic community, but aims to propose a solution for the provision of services in a social network environment supporting communities or organizations. In this case, multiple roles are identified for the community members and e-services may be provided based on the collaboration of specific members according to their role (Lewis, 2006; Oreilly, 2007; Vossen & Hagemann, 2007). For example, in the academic community students may be served by the Admission Office personnel to complete specific tasks, such as an application for a grade certificate. This should be feasible in the academic social network as well.

E-government service provision utilizing social networking technology could also be extended to include multiple organizations or service provision authorities, taking part in the same social network, each having a specific role. For example, in the academic social network, services provided by the Ministry of Economy could be integrated to provide taxation data to students applying for scholarships or specific benefits based on their family income.

Advantages of the aforementioned approach, compared to traditional e-administration service provision, include the integration of e-administration services from different sources in a unified environment. Moreover, the interaction model utilized to provide such services is familiar to users, encouraging them to adopt the proposed framework. Finally, the proposed model contributes to the vision of Web 2.0, where participants use social medial not only for informational purposes but also for transactional service provision.

The rest of the paper is organized as follows: Section 2 presents related work in the area of collaborative community support through social networks. Section 3 describes the main characteristics of the proposed collaboration model, as well as the gadget coordination life-cycle mechanism. Section 4 briefly describes the implementation of the implemented academic social network platform, called Unity, while Section 5

provides an example for e-administration using the proposed collaboration model. Conclusions and future directions reside in Section 6.

2 Related Work

Business Process Management utilizing social networking concepts has recently gained momentum, due to social software characteristics such as weak ties and mutual service provision, which fulfill requirements of collaborative environments (Bruno et al, 2011).

The possibility of collaboration using social networking infrastructure has been explored for specific communities, such as healthcare/medicine (Boulos & Wheeler, 2007), learning/pedagogical (Hiltz, 1998; McLoughlin & Lee, 2007) and academic (Bermejo et al, 2012). Results are encouraging, as they indicate that novel technological concepts, such as the ones offered through social networking sites, tend to attract users and facilitate interaction. Collaboration through task coordination in such environments can significantly facilitate e-participation and e-administration in the form of service provision.

Organizations and companies, such as IBM, wishing to promote collaboration between their members, have been researching how social networks affect intra-organization interaction, serving the Enterprise 2.0 vision. As a result, they have embraced social networking through public social networks, such as the aforementioned ones, by creating groups with their members as participants. However, as security and privacy issues emerge, many organizations employ private social networks; for example IBM have created the Beehive research project (Geyer et al, 2008). Private social networks, featuring similar functionality with public ones, can be established using existing social network development platforms, such as Elgg (Elgg). Companies encourage their employees to use their enterprise social networks so they can connect with other employees, help people socialize when they take a break, or even help contribute to other work related issues (DiMicco et al 2008), leading to new forms of business interactions and the notion of Enterprise 2.0.

Currently, collaboration within an organization through a social network remains mostly at an informational or communicational level; that is, the social network infrastructure is used only for exchanging information or performing trivial tasks, such as arranging a meeting. Experimental efforts have attempted to provide enhanced functionality to assist collaboration, such as file sharing (Shami, Muller & Millen, 2011).

Other works, such as (Bruno, 2012), (Hoegg et al, 2006) and (Ploderer, Howard & Thomas, 2010), explore how services offered by existing social networks can be utilized to promote collaboration between their participants. Moreover, the application of business models through social networks is also examined (Richter & Riemer, 2009). However, the aforementioned research efforts attempt to adjust collaboration requirements to the existing social network models and infrastructure, instead of proposing extensions to social networking models, which would accommodate interaction.

Existing social networking platforms used to establish either public or private social networks serving collaborative communities do not discriminate any participant roles. Moreover, they offer either an established model of predefined relationships which cannot be altered, or resort to an entirely user-defined model of relationships. In both cases, there is a need for the development of a new interaction model and an underlying social network implementation platform supporting collaborative communities, featuring different participant roles and relations and enabling collaborative task coordination.

3 Performing Collaborative Tasks in a Social Network Environment

3.1 Collaboration and Task Coordination to provide e-services

Collaboration in a typical social network is performed through exchange of information and notifications in a distributed fashion. In addition to sharing content and notifications through discrete streams and groups, the social network model should also support the provision of specific e-services, simple or complex, and enable its participants to complete corresponding activities in collaboration with other participants to provide these services.

These services may be provided by co-operating applications executed on a specific participant profile, authorized to complete the corresponding activity. Typical social networks enable applications, usually named gadgets, written in Javascript, to be executed on the user profile. These applications usually read data from the user profile and may invoke external applications through a web service interface. They also have access to store data in the user profile. In order to ask for services rather than information from another participant, a more sophisticated communication mechanism is required, facilitating information exchange between applications executed on different profiles.

We propose to treat services as tasks consisting of specific steps (e.g. activities) which may be performed by a specific role or roles and may involve the invocation of external services to be completed. Each activity corresponding to a specific task step is handled as an application, or gadget, which may only be executed in the profile of a participant having the proper role.

In order for collaborative tasks to be supported, inter-gadget communication executed in different profiles must be enabled. Based on available social network technology, gadgets may access and store data in a specific area of the profile they are executed on, called Application Data. In the proposed model, gadgets may share access to Application Data stored in the profile they are executed on, but also in external profiles as well, under certain conditions. Whenever there is need for inter-gadget communication, the sender-gadget updates this data, and the receiver-gadget can read the updates. Only when all input data is available, the receiver-gadget is allowed to start its execution. While the task is progressing, proper notifications are issued to collaborating participants.

3.2 Gadget recommendation & composition through planning

Each gadget, as any other program, needs specific input data to start its execution and when executed, produces output data. The co-ordination of tasks, e.g. the conditions under which specific activities may be executed, is performed based on the available input data of gadgets implementing the specific activities. A gadget implementing a specific activity cannot start its execution until all its input data are available. This data may be part of the user data stored in the profile the gadget is executed or produced as output data of other gadgets, which may be executed in the same profile, e.g. by the same user, or more frequently in external profiles corresponding to users having the proper role to invoke those gadgets.

As the number of available gadgets increases, an automated mechanism is required in order to perform gadget input-to-output matching and determine the composition of gadgets that produces the desired functionality. The proposed approach suggests the use of AI planning as a mechanism for automated gadget coordination, by formulating a gadget composition problem and representing it as a planning problem.

A planning problem is usually modeled according to STRIPS (Stanford Research Institute Planning System) notation (Fikes & Nilsson, 1971). A planning problem in STRIPS is a tuple $<I,A,G>$ where I is the initial state, A is a set of available actions and G is a set of goals. States are represented as sets of atomic facts. Set A contains all the actions that can be used to modify states. Each action A_i has three lists of facts containing the preconditions of A_i, the facts that are added to the state and the facts that are deleted from the state, noted as $prec(A_i)$, $add(A_i)$ and $del(A_i)$ respectively. An action A_i is applicable to a state S if $prec(A_i) \subseteq S$. If A_i is applied to S, the successor state S' is calculated as $S' = S - del(A_i) \cup add(A_i)$. The solution to a planning problem (plan) is a sequence of actions, which, if applied to I, lead to a state S' such that $S' \supseteq G$.

The representation of the gadget composition problem to a planning problem can be performed applying the following rules:

- The set of all available inputs that the user can provide to the social network formulates the initial state I of the planning problem.
- The set of all available outputs that the user wishes to receive by the desired functionality formulates the goal state G of the planning problem.
- The set of all available gadgets in the social network formulates the set A of actions. More specifically, each gadget is transformed into an action; the inputs of the gadget serve as the preconditions of the action, while the outputs of the gadget serve as the results of the action.

The planning problem can then be forwarded to external planning systems in order to acquire solutions, as the one presented in (Hatzi et. al, 2010). The produced plan will enable to determine the combination of gadgets that can be executed to perform the requested collaborative task.

3.3 Gadget life-cycle & execution

In order to develop and incorporate gadgets with enhanced functionality in the social network platform, a gadget lifecycle mechanism was developed, focusing on the following tasks:

- Each participant gains access to a list of available gadgets, depending on their role. The participants may install any of these applications on their profile.
- Whenever a participant installs a new gadget on their profile, a new set of AppData, specific for this participant and this gadget instance is created to store information related to this gadget instance. To complete a specific task, other cooperating gadget may update or access this data, enabling a "conversation" between two or more of them. AppData is used in the case of application collaboration as a common workspace or whiteboard, where all gadget participating in a task can use for data exchange.
- In order to provide enhanced functionality, gadget composition is enabled. Participants may demand a functionality model, by describing its available inputs and desired outputs, which will consequently be used by the mechanism described in Section 3.2 for providing a gadget composition plan that fulfills these requirements. The corresponding recommendation mechanism will prompt for the installation of gadgets that are included in the composition but are not yet installed on the participant profile.
- The participant may delete any of the previously deployed gadgets from their profile. Upon uninstall, the AppData associated with this particular participant and gadget is also eliminated.

During the execution of a gadget, the following distinct steps can be identified:

1. The gadget checks if all required inputs are available, by checking the values of the required AppData items.
2. If not, the recommendation mechanism is invoked and the necessary gadgets providing this data as outputs are identified. These gadgets may involve external applications. The user is notified accordingly.
3. The gadget requests the remaining required data from the user as inputs.
4. When all mandatory input data is available, the gadget is executed. Execution may include communication with external systems.
5. The gadget produces output data, by updating the appropriate AppData items.

4 The Unity Academic Social Network

In order to provide a concrete reference example where the proposed approach can be applied, the case of the academic community was considered, resulting in the development of the Unity academic social network (Nikolaidou et al, 2012). Unity implementation is based on the OpenSocial API, which is properly extended to facilitate participant roles and task co-ordination. A screenshot of the interface of the academic social network constructed using the Unity platform is depicted in Figure 1.

4.1 Participant Roles & Supported Relations

Members of an academic community include faculty members and additional teaching staff, undergraduate and postgraduate students, PhD candidates, researchers, administrative and technical staff. Each of them has specific responsibilities in the community, may represent specific service provision authorities, as for the University Library or the Student Admission Office, and may perform specific tasks to serve other community members. Member tasks and responsibilities are predetermined by their role in the community. Based on their role, members may take advantage of the predetermined relations in the academic environment and co-operate with others to accomplish specific tasks. Provided services in many cases concern administrative tasks; using the proposed interaction model they can be performed in a paperless way, promoting e-administration.

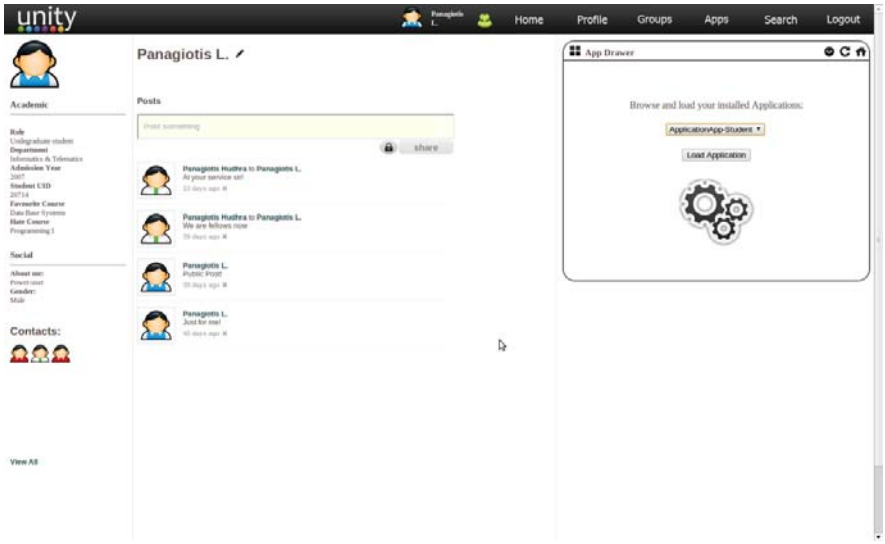

Figure 1. Screenshot of an example profile of the academic social network.

The discrete general roles defined in the proposed model that a participant of the academic social network can belong to are the following:

- Student: including undergraduate students, postgraduate students and PhD candidates
- Teaching staff: includes faculty members and additional teaching staff
- Administrative staff: includes secretariat employees, library employees, Erasmus office employees, Rector's office employees and all the rest of the University employees that could potentially provide services to community members.

Supported academic relations include:

- Tutor: a unidirectional relation declaring that a student is being taught / consulted by a member of the teaching staff. When this relation exists, the student benefits from specific services provided by the teaching staff member, as for example ask for a recommendation letter, or even co-operate with them to accomplish a task, for example the submission of a degree thesis assignment application, supervised by the teaching staff member, to the Student Admission Office.
- Facilitator: a unidirectional relation declaring that a community member is served by a member of the administrative staff. When this relation exists, the community member is the recipient of services provided by the academic staff member.

The social aspect of the network is not dismissed, therefore, the model also defines the social relation Fellow, denoting that two participants are socially connected, regardless of their roles.

4.2 The OpenSocial Framework & Apache Shindig

OpenSocial (OpenSocial) framework is a set of APIs for implementing applications for interoperable social networks. These social networks, known as OpenSocial containers, allow OpenSocial Gadgets to access information stored within the social network platform. Gadgets are built using OpenSocial APIs, which expose methods for accessing social data, application data and activities, within the context of a container. The same OpenSocial Gadget can run on more than one containers, e.g. a social network platform such as iGoogle or MySpace.

Apache Shindig (Apache Shindig) is an open source OpenSocial container and provides a reference implementation of the OpenSocial specification. Shindig processes JSON RPCs generated by gadgets, according to OpenSocial upper level API. For example, the following call retrieves information about all friends of a the participant with ID guid:

```
osapi.people.get({userId:'guid',groupId:'@friends'});
```

Such calls are processed by Handlers, implementing calls available to the social network developers. OpenSocial database is accessed by low-level calls, tightly depended on OpenSocial database schema. Shindig implements the corresponding class, which retrieves these elements through JPQL queries to the database.

For the implementation of the Unity platform, the Java version of Shindig was used and extended according to the academic community collaboration model. To do so, existing OpenSocial API calls had to be extended to support enhanced functionality.

4.3 OpenSocial Extensions

OpenSocial API was extended to accommodate different participant roles and relations, inter-gadget communication and an enhanced notification mechanism incorporating the concept of roles and specific participant notification (Nikolaidou et.al, 2012).

Extensions to OpenSocial include extensions both to the underlying database as well as to the corresponding API calls. Extensions in the lower level API, which is tightly depended on the database schema, are necessary to accommodate all database extensions. Some of them must also be propagated to the upper level API to support either additional parameters or parameter values of existing OpenSocial calls, used for gadget development (JSON-RPC interface) or to program the OpenSocial Container (REST interface).

Moreover, custom calls were implemented in Java, for functionality that is not defined through the OpenSocial API. Such functionality mainly concerns the creation of social network entities, such as users and connections, as this is a procedure not inherent to the management of the social network model, but specific to each implementation platform, while it is not related to gadget development.

4.4 Extending Unity Platform to support E-Administration Services

In order to support e-administration services as collaborative tasks, the Unity platform was extended to incorporate the recommendation mechanism defined in section 3.2 and the gadget life-cycle presented in section 3.3, resulting in the architecture presented in Figure 2.

More specifically, the faded parts were supported in the framework presented in (Nikolaidou et al, 2012); while the bright parts are the necessary components added to support e-administration services.

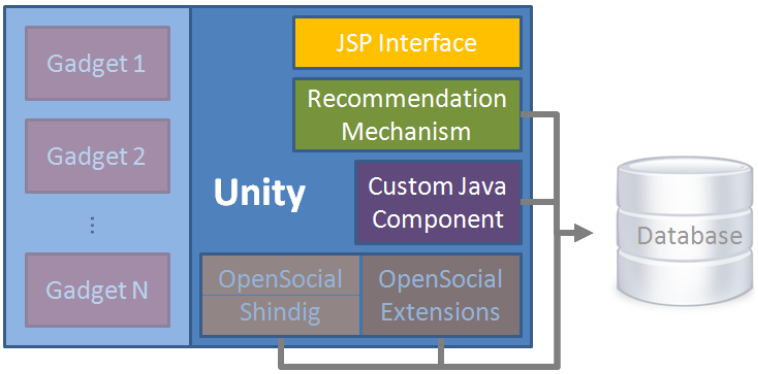

Figure 2. Unity framework architecture.

The *JSP Interface* was extended to accommodate gadget life-cycle. Participants may add gadgets in their profile based on their role in the academic community, while, when a gadget is added, corresponding AppData should be created for the specific participant. Furthermore, prior the execution of a gadget all necessary input data should be available. Such restrictions are managed by the supported JSP Interface, for all gadget in a unified fashion. To provide such functionality, the OpenSocial database is further extended to accommodate a Gadget Registry. Additional custom calls (not included in OpenSocial API) were created for this purpose, grouped in *Custom Java Component.*

The *Recommendation Mechanism* accommodates the provision of complex e-administration services by facilitating the combination of gadgets. It is invoked by the participants through the JSP interface, responsible to present them with the appropriate gadget compositions. The recommendation mechanism transforms the gadget composition problem to a planning problem and uses external planning systems (Hatzi et. al., 2010) to acquire solutions, i.e. plans that indicate how available gadgets can be composed to achieve enhanced functionality. In the same fashion, during gadget execution, the recommendation mechanism can indicate the prerequisites for specific gadgets, that is, the gadgets that should be executed and completed successfully before this gadget obtains all necessary inputs and is able to start its execution.

5 E-administration service example

As an example of e-administration service consider the Thesis Assignment task. Students accomplish this task by filling out a corresponding Thesis Application Form, which is submitted to the corresponding professor. If he/she agrees, the form is signed and the supervising committee is indicated. If the request is denied, the student receives the form back. If the request is accepted, the professor forwards the form to the Student Admission Office to confirm the thesis assignment, taking into account other obligations the student may have and properly notify both the student and the professor (preferable by e-mail). A BPMN diagram for the Thesis Assignment process is depicted in Figure 3. As indicated in the figure, corresponding data are filled by the student and the professor, if the student request is accepted. The first activity (Apply for Thesis) is performed by the student, the second and third by the corresponding professor, indicated in the Thesis Application Form, and confirmations by Admission Office personnel.

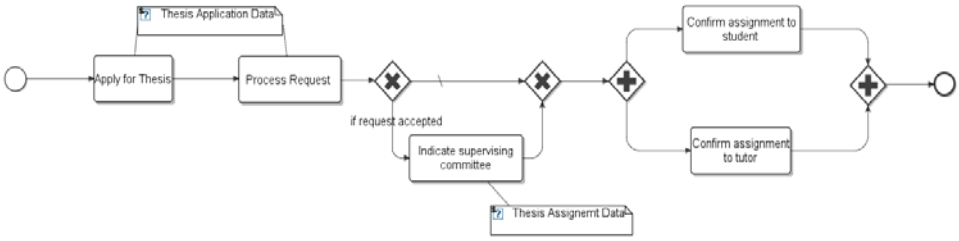

Figure 3. Thesis assignment BPMN diagram.

In the context of the academic social network, the student, the professor and admission staff must be able to perform the corresponding activities using gadgets installed and executed on their profiles.

It should be noted that the gadgets composing a specific task and the way gadgets are coordinated using specific application data fields are specified by the gadget developer, while Unity platform provides the necessary features to make this possible. Participant collaboration is possible through the proposed notification mechanism;

each gadget can issue notifications targeted to a specific participant or participants having a specific role that need to take action next.

The process is initiated by the student, who selects the corresponding gadget, named "AssignThesis-Student", from the gadget drawer and installs it, as depicted in Figure 4 (left). If the student attempts to execute the gadget before any of thesis subjects have been announced, an informative message is displayed (Figure 4 – right).

Upon installation, an AppData item called *thesis_status* is added in the application data table for the specific application instance for the specific user, "Sample Student". Communication among the coordinating gadgets for the completion of the Thesis Assignment task is possible, since all of them are allowed to access and modify the AppData items created by collaborating gadgets on different profiles.

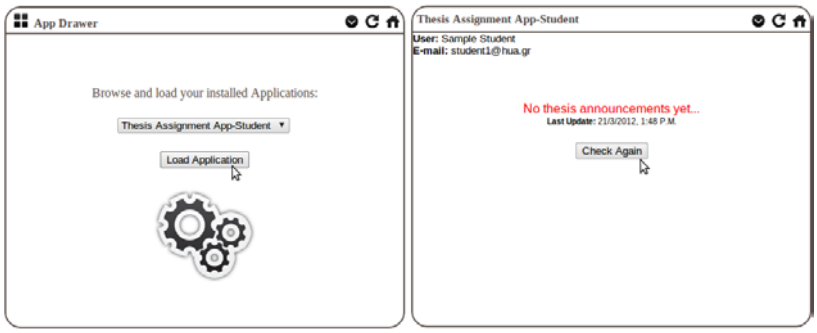

Figure 4. Thesis Assignment task: Student installs and executes gadget.

When thesis subjects are published by the teaching staff of the Department, the student is notified and the "AssignThesis-Student" gadget retrieves them and enables the student to select a subject and request the assignment, as depicted in Figure 5, changing the value of the *thesis_status* AppData item to *pending*.

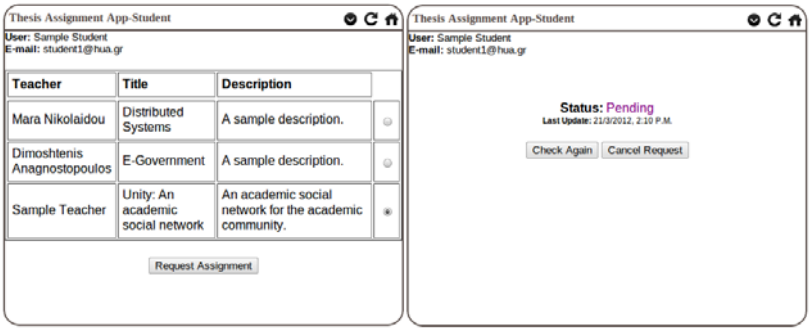

Figure 5. Thesis Assignment task: Student requests assignment.

The selected subject in this example is supervised by "Sample Teacher". The corresponding gadget checks whether "Sample Teacher" is a tutor of "Sample Student" and issues a notification targeted to the specific professor, as depicted in Figure 6, to

let him know of the request and to inform him that he must be involved in the task by installing the corresponding gadget and taking some action.

After the Professor is notified of "Sample Student" request (Figure 5), he must install and execute the corresponding tutor gadget, named "AssignThesis-Teacher". Upon installation, the following appropriate AppData items are generated in the professor's profile: *thesis_title*, *thesis_description*, *student*, *supervisors*. The student's gadget can access and modify these data, in order to achieve inter-gadget communication.

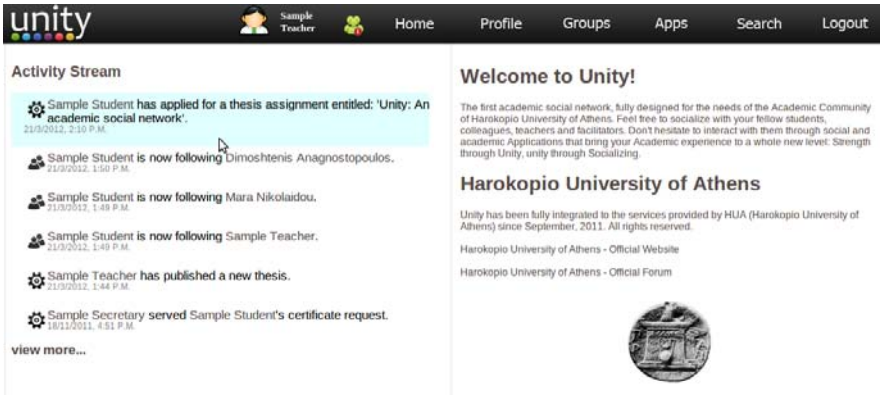

Figure 6. Thesis Assignment task: Professor is notified to be involved in the task.

When executing the gadget, the professor views all pending requests, accepts the request of the specific student, indicates the supervising committee members as depicted in Figure 7.

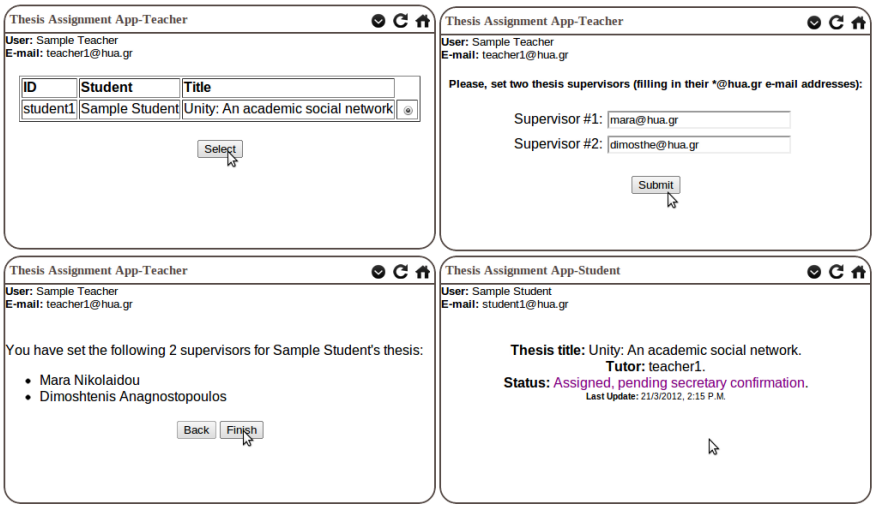

Figure 7. Thesis Assignment task: Professor views pending requests, accepts and indicates supervisor committee.

The corresponding AppData, some residing on the professor and some on the student profile, are updated. For example, the *thesis_status* item on the student profile is changed to *awaiting_conformation*. The student and Admission Office staff members are notified for his actions. The professor gadget notifies the student of the progress, and also notifies the Admission Office staff to be involved, by executing its own gadget. The Admission Office staff performs the necessary checks and confirms the assignment, thus completing the Thesis Assignment task, as depicted in Figure 8.

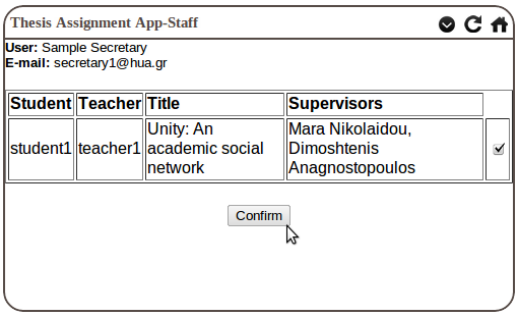

Figure 8. Admission Office staff confirms thesis assignment.

Upon task completion, the Admission Office gadget updates the AppData on the student profile, to indicate the status of the thesis as assigned, and confirmation notifications are sent to both the student and the professor, as depicted in Figure 9.

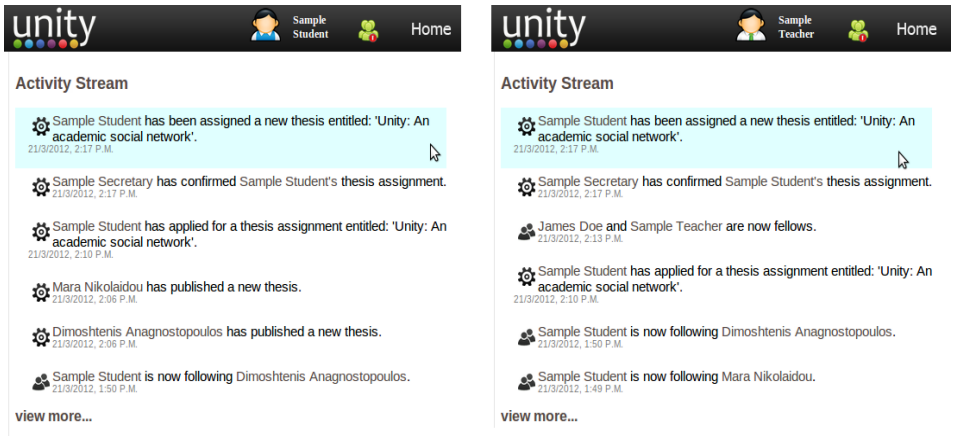

Figure 9. Notifications on student and professor profiles.

6 Conclusions & Future Work

This paper presented an approach supporting the provision of e-administration services through a novel collaboration model, employing social networking technology.

Interactions between participants providing and requesting services were modeled as a social network community, featuring extended participant roles and relations. A specific predefined role was assigned to each participant, denoting their obligations and responsibilities in the community. Based on these roles, specific relations were defined, denoting the allowed interactions that can take place between participants. Service provision is performed through collaborating gadgets; each gadget can only be executed by specific participant roles. Collaboration in many cases requires matching between gadget inputs and outputs. A recommendation mechanism, using planning, is utilized to perform the matching and derive an execution plan to facilitate coordination.

The proposed e-service provision model was implemented within Unity academic social network, which already supported discrete participant roles and relation between them, enabling the assignment of specific activities to specific roles or participants. Unity framework was implemented by extending OpenSocial framework, based on Apache Shindig. To support the collaborative provision of e-administration services, the Unity framework was further extended to accommodate gadget lif-cycle and a recommendation mechanism.

Pilot e-administration services were developed and tested, such as Thesis Assignment, presented as an example, and Student Restaurant Card. Experience gained by supporting such e-administration services through collaborating tasks produced encouraging results. We continue developing and testing more complex services, which require coordination not only with University authorities, but also with external authorities, for example taxation e-gov services.

The proposed extensions in social network collaboration model provides a social networking paradigm that can be utilized to serve e-administration purposes not only in the academic environment, as presented in this paper, but may also be applied to other collaborative communities featuring participant roles and complex relations between them, enabling task coordination and thus service provision based on these roles. The integration of the proposed interaction model with e-government services lies among our future goals.

7 References

1. Bermejo J. A. A., Bravo, C. B., Mateos, M. J. R., Piera, J. R., (2012). Social Networks in the Higher Education Framework - Understanding the University as an Organization: Inlumine, Our Study Case. Handbook of Research on Business Social Networking: Organizational, Managerial, and Technological Dimensions, IGI.
2. Boulos, M. K., Wheeler, S., (2007). The emerging Web 2.0 social software: an enabling suite of sociable technologies in health and health care education. Health Information & Libraries Journal, Volume 24, Issue 1, pages 2–23.
3. Bruno, G., (2012). An Approach to Defining Social Processes Based on Social Networks. Handbook of Research on Business Social Networking: Organizational, Managerial, and Technological Dimensions, IGI.

4. Bruno, G., Dengler, F., Jennings, B., Khalaf, R., Nurcan, S., Prilla, M., Sarini, M., Schmidt, R., Silva, R., (2011). Key challenges for enabling agile BPM with social software, J. Softw. Maint. Evol.: Res. Pract.; 23:297–326
5. Dais, A., Nikolaidou, M., Anagnostopoulos, D., (2011). OpenSocialGov: A Web 2.0 Environment for Governmental E-Service Delivery. EGOVIS 2011: 173-183
6. DiMicco, J., Millen, D. R., Geyer, W., Dugan, C., Brownholtz, B. and Muller, M., (2008). Motivations for Social Networking at Work. CSCW '08 Proceedings of the 2008 ACM conference on Computer supported cooperative work,ACM New York, NY, USA.
7. Elgg, Open Source Social Networking Engine, http://elgg.org/ [accessed 20/02/12]
8. Hatzi O., Vrakas D., Bassiliades N., Anagnostopoulos D., Vlahavas I. (2010). A Visual Programming System for Automated Problem Solving. Expert Systems With Applications, Elsevier, Vol. 37 (6), pp. 4611-4625.
9. Fikes, R., Nilsson, N. J., 1971. STRIPS: A new approach to the application of theorem proving to problem solving, Artificial Intelligence, Vol 2 (1971), pp 189-208.
10. Geyer, W., Dugan, C., DiMicco, J., Millen, D. R., Brownholtz, B. and Muller, M., (2008). Use and Reuse of Shared Lists as a Social Content Type. CHI '08 Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems, ACM New York, NY, USA
11. Hiltz, S. R., (1998). Collaborative Learning in Asynchronous Learning Networks: Building Learning Communities, 3rd WebNet World Conference of the WWW, Internet, and Intranet Proceedings.
12. Hoegg, R., Martignoni, R., Meckel, M., Stanoevska-Slabeva, K., (2006). Overview of business models for Web 2.0 communities, Proc. Workshop Gemeinschaften in Neuen Medien GeNeMe Dresden: TUDPress, Dresden, p. 33-49
13. Lewis, D., (2006). What is web 2.0?, ACM Crossroads 13, Vol. 13, Issue 1, pp. 3-3.
14. McLoughlin, C., Lee, M. J., (2007). Social software and participatory learning: Pedagogical choices with technology affordances in the Web 2.0 era, In Proceedings ASCILITE Singapore 2007.
15. Nikolaidou, M., Hatzi, O., Katsivelis-Perakis, P., Hudhra, V., Anagnostopoulos, D., (2012). Utilizing Social Network Technology to Support Collaborative Tasks. Computer Supported Cooperative Work, Springer, *under review*.
16. Oreilly, T., (2007). What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. Communications & Strategies, No. 1, p. 17.
17. Ploderer, B., Howard, S., Thomas, P., (2010). Collaboration on Social Network Sites: Amateurs, Professionals and Celebrities, Computer Supported Cooperative Work (CSCW), Volume 19, Number 5, 419-455.
18. Richter, A. and Riemer, K., (2009). Corporate Social Networking Sites – Modes of Use and Appropriation through Co-Evolution. ACIS 2009 Proceedings.
19. Shami, N. S., Muller, M. J., Millen, D. R., (2011). Browse and discover: social file sharing in the enterprise. CSCW '11 Proceedings of the ACM 2011 conference on Computer supported cooperative work. pp. 295-304.
20. Vossen, G., Hagemann, S., (2007). Unleashing Web 2.0: From concepts to creativity. Ubiquity 2007, December, Article 3.