# *Collaborative Management of Applications in Enterprise Social Networks*

Ourania Hatzi, Giannis Meletakis, Mara Nikolaidou, Dimosthenis Anagnostopoulos
Department of Informatics and Telematics
Harokopio University of Athens
Athens, Greece
{raniah, meletakis, mara, dimosthe}@hua.gr

***Abstract*—Enterprise social networks are gaining momentum as a platform for collaboration between members of an enterprise, leading to the notion of Enterprise 2.0. Participant collaboration includes communication, information and data dissemination and also application execution, leading to task completion. This paper presents an approach extending social network data model to promote participant collaboration through collaborative application execution and management. The proposed extensions support not only process monitoring but actual execution, management and composition of applications within the context of the social network, in a uniform way. In order to achieve this effectively, external services must be wrapped and integrated in the social network environment as applications, and consequently they must be mapped to business tasks executed in each participant profile, according to their actual role and responsibilities in the enterprise. The social network framework must take into account the actual roles, relationships and responsibilities of the members of an enterprise, which, combined, dictate enterprise workflow schemes and policies. To explore the potential of the proposed approach, a prototype social network platform is discussed as a case study.**

***Keywords— Social BPM; Enterprise 2.0; social collaboration; task coordination; task completion; service provision; application execution; roles; relationships.***

## I. Introduction

Social networks have been established as a prominent model for communication and interaction between individuals, as well as among members of communities or organizations [1]. Currently, there are numerous social network platforms, both general-purpose, such as Facebook, and targeted to specific communities, such as Myspace. Social network platforms enable user communication in everyday social life and they continuously offer enhanced functionality and advanced features including application execution through external service invocation and integration with other social networks [25][26][29][8].

The possibility of collaboration using social networking infrastructure, either public or private, has been explored for specific communities, such as healthcare/medicine [6], learning/pedagogical [21][30] and academic [4]. 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.

A common phenomenon in such cases is the exploitation of typical features of existing social networks, using them for a different purpose than the one originally intended. For example, users in many cases exploit privacy settings to increase visibility of certain items rather than decrease it, or use groups to establish a reputation mechanism [36][42].

Corporations and organizations have incorporated social network technology, either by using popular social networks [41] or more often by developing private enterprise social networks [15] aiming at more effective knowledge dissemination, intra-organization communication and efficient collaboration between their members [17].

Collaboration within an organization, utilizing enterprise social networks, remains mostly at the informational or communicational level; that is, the social network infrastructure is used only for exchanging information, performing trivial tasks, such as arranging a meeting, or even share and collaboratively edit documents [39].

However, to further promote the utilization of enterprise social networks in organizations, their contribution in performing every-day activities and complete business goals should be strengthen [24]. Creating an enterprise social network environment that will effectively merge knowledge gathering capabilities through social collaboration with task accomplishment features is not straightforward [27]. However, it may prove promising for supporting employees in being more agile and efficient and consequently enhancing organizational work. This potential has already been discerned by the research community, as revealed by the recently established field called Social BPM [40][38], which investigates the adoption of social features in the Business Process Management discipline.

Collaboration to perform a task or complete a goal is usually performed by communicating with others, sharing information or executing applications. The first two, are basic characteristics of social networking technology already supported in enterprise social networks. Although single application execution is a standard feature in a typical social network, application interaction by exchanging or sharing data is not facilitated. To enable application execution to serve a mean for collaboration between participants, application interaction and combination features should also be supported by enterprise social network platforms.

More specifically, participants should be engaged to perform specific activities by executing the corresponding applications, according to their role in the organization and current circumstances, and collaborate with others based on business policies and rules. The underlying social network framework must provide ways to define such a role-based application execution mechanism, while it should also accommodate for specific, more fine-grained responsibilities of certain participants. At the same time, it should facilitate the management and discovery of cooperating applications in a uniform, generic and reusable way; this implies that the application management mechanism should be data driven and as independent from specific applications as possible.

In this paper we discuss extensions to the typical social network data model [44] to support and facilitate collaboration which eventually leads to the notion of Social Business Process Management [10]. These extensions focus on application interaction, collaborative application combination to accomplish a task and collaborative application management necessary to achieve application combination. Application execution should be restricted by each participant responsibilities in an organization, thus business policies and rules should also be somehow represented in the extended social network data model. Incorporating such functionality in an enterprise social network can promote the notion of Enterprise 2.0, since business processes may be completed in the social network environment, incorporating all the proposed features. Such functionality is not limited to accommodating Enterprise 2.0 but can also be utilized within the wider view of Organization 2.0; not only enterprises but also different kinds of collaborative communities such as researchers or volunteers can benefit a well [7].

The authors have already proposed extensions to the social network data model to enable role-based collaboration between members of collaborative communities or organizations, and consequently developed the Unity Social Networking Platform to support these extensions, serving as the enterprise social network in a University setting, as discussed in [19]. The Unity platform supports specific roles and relations found in collaborative communities, in order to reflect the rights and responsibilities of each member, which emerge from his position in the community. Enhanced functionality, incorporating services offered within the social network as well as by external systems, is provided through application execution; participants execute applications on their profiles and can collaborate with each other as applications can read and write data from a shared profile data workspace. This paper further elaborates the ideas presented [19] by proposing a data driven application execution mechanism enabling application composition, as well as a collaborative application management mechanism enabling application registration and administration. To this end, the role / relation interaction and collaboration model proposed in [19] is further extended. Unity framework current version, under the working name CollaTo (Collaboration Tool) project, can be found at https://github.com/meletakis/collato.

The rest of the paper is organized as follows: Section 2 presents related work in the area of private social networks to facilitate Enterprise 2.0. Section 3 provides a brief overview of the Unity social network, which served as the platform to integrate the proposed features for data driven application execution and collaborative application management, while Section 4 focuses on the main contribution of this paper, which concerns social network data model extensions to support the proposed features. Experiences from testing the proposed framework are discussed in Section 5, while conclusions and future directions reside in Section 6.

## II. Related Work

The utilization of Web 2.0 technologies, within enterprises or organizations, to promote collaboration between organization members and knowledge sharing, consists the Enterprise 2.0 vision [23], Although the understanding of such a business model is still in some flux, the tools utilizing are rather clear. A prominent tool serving Enterprise 2.0 vision is enterprise social networks, aiming to explore how social network technology may enhance intra-organization collaboration.

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 help contribute to work related issues [13], leading to new forms of business interactions. Private social networks also tackle emerging security and privacy issues. The most well known example of such an enterprise social network, is the Beehive research project [15], created by IBM, and evolved into the IBM SocialBlue platform [9]. SocialBlue platform is also used as a test bed to test solutions supported by IBM social software toolkits.

Recent attempts to provide enhanced functionality to assist collaboration includes features, such as file sharing and collaborating file editing [39]. Such options are implemented as external services integrated with the social network platform.

Research works, such as [11][22][35], 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 [37][14][32]. In [5], the authors examine how the architectural principles behind BPMS and social software can be combined in order to develop a unified infrastructure supporting features of both software types. In [23] a set of guidelines is suggested for augmenting BPMS with social software features. In [9] BPMN 2.0 is extended to include social collaboration semantics. In [32] a wiki-based implementation of a workflow system is presented. In contrast to these efforts that are based on the typical BPM paradigm, in [31] a framework for supporting Adaptive Case Management in social networking environments is introduced. In [16] a service integrated into Facebook for managing personal workflows following a case-based reasoning approach is proposed.

The aforementioned research efforts attempt to adjust enterprise collaboration requirements to the existing social network interaction models and infrastructure, instead of proposing extensions to the social networking data model, which would accommodate enhanced collaboration features between enterprise members, respecting their role in the Enterprise and corresponding enterprise policies and rules. In contrast to popular social networks, enterprise network participants should interact based on predefined roles, and each participant is expected to contribute accordingly, completing certain tasks assigned to them. Thus, a requirement emerges for the development of a new collaboration model and underlying social network platforms supporting Enterprise 2.0, featuring complex interaction models, multiple member roles and collaborative task execution based on these roles [28][34][43][10][17].

## III. Unity Framework Overview

This section provides an short overview of Unity framework supporting the extended interaction model proposed in [19]. Unity aims at serving as an enterprise social network in a University setting.

In order to effectively serve Enterprise 2.0 or Organization 2.0 [7], a corresponding enterprise/organizational social network should accommodate:

- representation of discrete organization member roles
- incorporation of the organization co-operation model based on predefined relations
- information sharing and promotion of collaboration between organization members in a familiar yet intriguing way
- integration of services offered by external systems through a unified environment
- provision of services through application execution by specific organization members to others, based on their roles and relations

### *A. Participant Roles and Relationships*

A social network serving Organization 2.0 should feature discrete roles for participants, corresponding to their position and responsibilities on the organization. Roles determine possible relations between participants. The decision about how specific the roles should be is based on whether the specialization affects the emerging relations. Roles also determine additional data stored in the profile for each participant.

Both organization and social relations could be supported. Organization relations can be either unidirectional, indicating that an organization member receives services from another member, or bidirectional, indicating that the members cooperate to achieve certain tasks. When a relation exists, the object of the relation receives updates, posts and material published to the corresponding stream of the subject member profile, and benefits from specific services provided by them.

The social relation defined as fellow for example, corresponding to the generic relation offered by existing social networks, is a bidirectional relation denoting that two participants are socially connected. This relation may exist between any two participants, regardless of their roles; if the relation exists, each participant receives posts, updates and material published to the social stream of the other.

### *B. Streams*

The most common operation that a participant performs in a social network is publishing content, which can be of a variety of types, such as links, texts, files, multimedia etc. Published information is propagated in the form of a stream to all participants related to the publishing entity, who receive notifications and updates about the publication, urging them to review it and possibly contribute to it, as dictated by the notion of collaborative content in Web 2.0 [3].

In an organization, specific streams should be defined based on member roles and relations. Apart from the intra-organization member relations, the social aspect of the community should not be dismissed; therefore, each member may develop a social relation with any other member of the community, regardless of their roles in it. At the same time, a clear separation between them should be maintained, thus a more complex propagation mechanism is introduced incorporating more than one discrete streams.

The combination of discrete participant roles, multiple streams, extended relations and rules governing the propagation of content successfully achieves the separation between the organizational and social aspects of the organization.

### *C. Groups*

The combination of roles, relations and streams does not facilitate fine-grained content propagation; therefore, a more elaborate mechanism for content delivery is required, through groups. Groups are arbitrary sets of contacts that any social network member can create and modify dynamically. Each publication to a certain group belongs to a corresponding custom, ad-hoc group stream and is propagated to all members of this group.

### *D. Application Execution & Coordination*

Collaboration in a typical social network is performed through exchange of information and notifications in a distributed fashion [18]. In addition to sharing content and notifications through discrete streams and groups, Unity enables its participants to complete specific business tasks in collaboration with other participants [14], using interacting applications executed in specific participant profiles. Typical social networks enable applications, 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, but they are not allowed to store data on them. In order to ask for services rather than information from another participant, a more sophisticated communication mechanism is supported by Unity, facilitating information exchange between applications executed on different profiles.

Evidently, in order for collaborative tasks to be supported, inter-application communication executed on different profiles must be enabled. These applications may implement specific steps or activities contributing to the completion of the specific task. Each application, as any other program, needs specific input data to start execution and, when executed, produces output data. The coordination of tasks, e.g. the conditions under which specific activities may be executed, is performed based on the available input data of applications implementing the specific activities. An application 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 application is executed on, or produced as output data of other applications, which may be executed on the same profile, e.g. by the same participant, or more frequently on external profiles corresponding to participants having the proper role to invoke those applications.

Based on available social network technology, applications may access and store data in a specific area of the profile they are executed on, called Application Data [12]. In Unity, applications may share access to Application Data stored in the profile they are executed on, but also in external profiles, under certain conditions. Whenever there is need for inter-application communication, the sender-application updates this data, and the receiver-application can read the updates. Only when all input data is available, the receiver-application is allowed to start its execution. Note that this mechanism is private to application and invisible to participants. While the task is progressing, proper notifications are issued to collaborating participants. Whenever the participants need to be notified of activities that take place by application interaction, the application is responsible for declaring that explicitly. Obviously, the participants collaborating for a specific task must be properly associated with corresponding relations.

### *E. Drawbacks identified in Application Execution & Coordination*

Unity social network was employed on a prototype level as a platform for collaboration between students, faculty and administrative staff at the Department of Informatics and Telematics of Harokopio University of Athens. Integration with the University LDAP server enables the verification of participants and their role within the academic community. After using the platform for a year, the following conclusions were drawn:

- The support of discrete roles, namely students, faculty and administrative staff, and business/organization relations, such as tutor, a unidirectional relation defined between students and faculty members, in addition to social ones, such as friend, was well accepted by participants. They considered it a valuable additional feature for a enterprise social network, taken into account that when performing a specific task, such as diploma thesis selection, the role and responsibilities of each participant in the University community should be taken into account.

- The utilization of Application Data concept, supported by social network data model [33], to store application data was proven inefficient due to complexity and performance issues. Thus, an alternative approach to store and retrieve data produced by application executed within the social network should be explored.

- Participants are the only responsible for identifying the application they should execute in their profile to produce the necessary data to complete a task. Furthermore, when an application cannot be executed because its input data is missing, the participant must identify the proper application combination that must be executed to produce them. This might be trivial when the number of available applications is limited, but becomes more complex when their number increases and alternative combinations may be valid. Thus, a recommendation mechanism should be supported, utilizing an application registry.

- Participants should be ensured that all the applications available to them can be actually executed. To do so a registry of available applications and also valid application data, exchanged between applications, should be maintained. Furthermore, application developers should be assisted in registering their applications and identified their input and output data. While the second seams a straightforward task, the first one is strongly related to the application data registered by other developers. It is possible that an application may not have all the input data needed for each execution. Would it be possible to register such an application? Thus, a collaborative application management mechanism should be supported for application developers.

To summarize, while the notion of supporting discrete roles and relations to depict real world constraints in intra-organizational interaction was proven successful in practice, the application execution and coordination mechanism proposed to accomplish tasks needs further improvements.

## IV. Collaborative Application Execution and Management in a Social Network Environment

To overcome the drawbacks identified in the previous section, the social network data model was further extended to support:

1. The efficient data driven application execution enabling application combination. To provide such a mechanism a dictionary for both applications and application data should be supported. The supported collaboration model was also further extended to enable fine-grain application execution management.

2. The collaborative management of applications by application developers enabling application and application data registration.

### *A. Extended Collaboration Model*

The extended collaboration model of Unity Framework enables coarse-grained control over application execution rights, based on actual participant roles and relations in the organization or enterprise. More specifically, it allowed for the specification of restrictions concerning the ability of certain authorities to perform certain tasks. This was achieved by specifying that each participant belonged to a specific role, and specific applications were able to be executed only on participant profiles bearing a specific role.

This mechanism did not allow fine-grained application execution control; the extensions proposed hereby attempt to resolve this issue. Consider a case when one or two particular individuals in an organization are authorized, perhaps for a short period of time; to perform a specific task on behalf of the organization, and this does not emerge from their general roles - in fact they may even have different roles in the organization - but accommodates a temporary organization need; for example an individual and his substitute could be the legal representative of a company who is authorized to negotiate and sign a specific contract on behalf of the company.

To facilitate such cases, we propose a more elaborate interaction model in which application execution control is based on an intermediate abstract notion, instead of being directly connected to the social network structure. The proposed model introduces the notion of *responsibility*; which represents the right of either a specific individual participant or of a specific category of participants corresponding to a certain role in the organization, to perform a specific discrete task.

In the proposed social network data model extension, application execution is controlled by responsibilities; each application can be executed only by participants having a certain responsibility assigned to them. Responsibility assignment can be performed either on specific participants or on entire participant roles, in a uniform fashion. The Python/Django implementation framework proved very suitable to implement such feature, as it provides the *django_content_type* element, which can exploit the database structure into accepting multiple data types for a specific field, without affecting its canonical form. The corresponding entities are included in the proposed extended social network data model, which is presented in Fig. 1, as an ER diagram.

Similarly to roles and relations, which are different for each social network constructed using the proposed Unity Framework, responsibilities are also not predefined and can be developed on demand, even after the launch of a social network, as new applications are introduced in it.

An additional advantage of the proposed extension concerns the cardinality of assignments. Before the extension the execution control model was not flexible enough, as each individual could belong to exactly one role in the organization and have the corresponding application execution rights assignments. The extended model allows for the assignment of multiple responsibilities to an individual; either directly to him or indirectly through their role in the organization.

### B. Data Driven Mechanism for Application Interaction

Refined application execution in the proposed framework is the first step to provide Social BPM features in enterprise social networks, as it enables the invocation of external services resulting in the completion of business tasks within the social network environment. The second and most important factor enabling collaboration, though, is collaborative application execution. This includes the ability to share and exchange content and data between applications, which was supported in the Unity Framework using Application Data concept. Application Data structure is in practice a group of data fields added in the participant's profile whenever the application producing them is added for execution in the profile as well. Thus, they represent a static data structure added in the profile before even the application is executed, waiting to be filled upon application execution. Their management proved to be complex, while there were performance issues in data access.

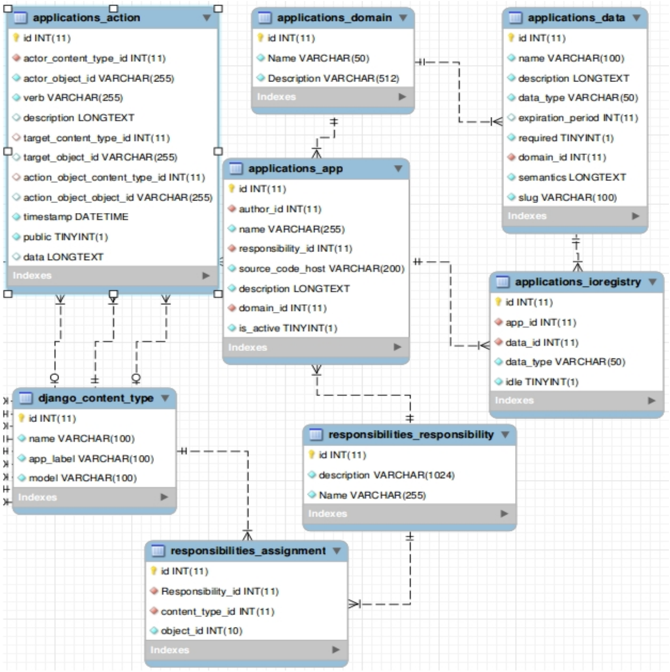


Fig. 1. Extended Social Network Data Model including Application Registry Entities and Responsibilities.

An alternative approach to store application data is to utilize activity steams [1] for this purpose. The concept of *Application Activity Stream* is introduced for application data storage and retrieval. In addition, it includes the ability of applications to cooperate, exchanging data without intervention of the user, which leads to the notion of application *composition*.

More specifically, *Application Activity Stream* was used for storing data produced from applications; in its simplest form, an activity consists of an *actor*, a *verb*, an *object* and a *target*. This format was extended to cover the requirements of the proposed data model by adding a *description* and *data* (see table *applications_action* of Fig. 1). The *actor* field is used to store the application name and the *object* field is used to store the user who executed the application. The *description* stores the name of the data which is produced by an application instance execution, while the *data* field stores the actual values of the data, using the JSON format. When some application, in order to be executed, requires as input data that is provided by another application as output, this application has to perform a search in the *description* field of application stream to identify if the required data exists.

### *C. Application Composition*

The previous version of the Framework did not provide an inherent composition mechanism; the authors' work in [20] was a first attempt to application composition within the Unity Framework, applying an idea emerging from previous work on web service composition, extracting the problem, applying AI Planning techniques, solving it in external systems and returning the solution to the social network framework. This idea was interesting and quite generic; however, it was not dynamic since it provides an overall solution of application composition.

In this paper we also discuss an inherent mechanism for application composition, fully adapted to the proposed framework, and independent of external systems. The proposed application composition mechanism is data driven; in the sense that the composition techniques for application matching are based on data exchanged between applications and not on applications themselves. This provides flexibility in application composition, as alternatives of applications are easily discovered and can be used as substitutes in cases when the user does not wish or is not allowed, based on their responsibilities, to execute specific applications.

The proposed mechanism requires that each application introduced in the social network is modeled through its inputs and outputs (see tables *applications_app*, *applications_data* and *applications_ioregistry* of Fig. 1). Inputs and outputs are represented as concepts with a unique identification scheme in the social network database; while developers register their applications in the social network, they have to indicate the concepts that their applications use as inputs and outputs. The first step in this procedure is to indicate whether the application being registered uses, either as inputs or as outputs, concepts that already exist in the specific social network; if so, the developer is asked to confirm that and a matching link is created as a dependency between the applications using the same concepts. If two applications both use the same concept as input, or both as output, then these applications are potential alternatives for a specific composition step. On the other hand, if an application uses a concept as input while another uses the same concept as output, these applications are potential consecutive steps in a composition. If such dependencies are not found, during an application registration, then the developer is asked to indicate the new concept that their application uses; these concepts are marked as idle and are later used for matching with new applications introduced in the social network.

The use of concepts instead of simply input or output data enables for the future incorporation of semantics in the social network. Semantics can potentially lead to improved compositions, as the terms used for describing application inputs and outputs are not required to be syntactically identical but they are only required to be semantically similar.

Apart from data exchanged between applications, contributing to application composition in the social network, applications also utilize data provided by users on demand, at execution time, or data found in the user profiles.

### *D. Collaborative Application Management*

The developer, when registering an application, is able to indicate which of the data used are user provided or profile data; such data are not included in the matching procedure for application composition in order to avoid errors and optimize performance.

Developers can collaboratively manage application interaction when they are registering their applications. Collaborative management ensures that the applications in the social network will be interoperable, without requiring a strict control over the application implementation and interface.

The developer while registering an application indicates that some of its inputs should be provided by other applications as outputs, interactions through the Application Activity Stream, as described in the previous section. If the corresponding applications are already registered in the social network, a dependency is added between them, indicating that an application receives data from another application; therefore, the execution of the application producing the data is a prerequisite for the execution of the application receiving the data.

If the corresponding applications are not already registered, the application at hand can still be registered; however, in this case, its inputs will be annotated as *idle*. This indicates that the application is not available for immediate execution because one or more of its prerequisite applications are missing; however; as more applications are added to the social network, the proposed framework performs the necessary input-to-output matching in order to suggest possible dependencies which the developers have to confirm or decline.

The dependencies discovered among applications result in the development of a graph of potentially interconnected applications, subgraphs of which are all possible compositions at any given time. In order to provide convenience to the user, this graph is visualized in an explorable way within the social network.

The introduced entities are depicted in the extended social network data model of Fig. 1, while the next section provides indicative screenshots of instances of application registration and the composition graph for an example case study.

## V. Case Study

The current version of Unity featuring collaborative application management is tested as intra-university private social network in Harokopio University of Athens. Besides exchanging social information participants may also put into effect the proposed features towards Social BPM. For this purpose, certain university business procedures were selected and provided as services through the social network. Such a procedure is the student graduation application.

In order for a student to be eligible to apply for graduation, the following requirements should be met:

- All courses have been successfully completed.

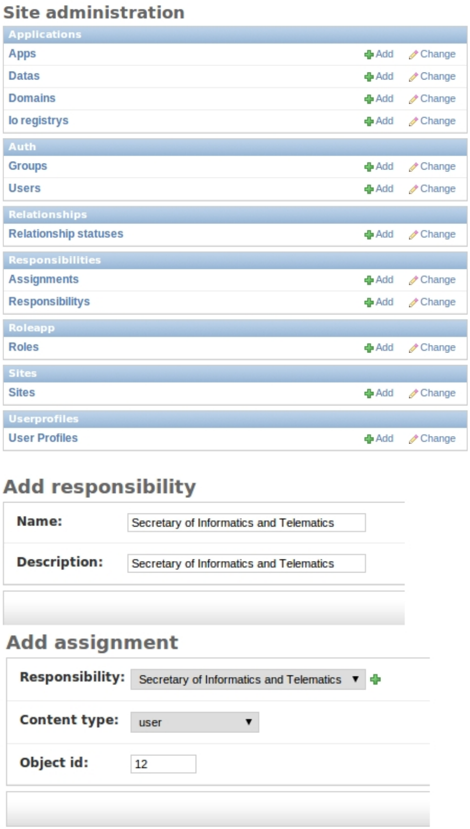
- The degree thesis has been submitted to the University Library.

- All books borrowed from the University Library have been returned.

- The student ID, transportation card and thesis certificate have been returned.

The student can subsequently fill out a graduation application form and submit it to the Department Secretariat, who confirms that all requirements are valid and notifies the student of the graduation ceremony date.

All aforementioned tasks are performed by the student by executing the corresponding applications on their profile; the Application Interaction mechanism in this case enables other stakeholders, such as a library officer or the department secretariat to be notified and perform the tasks assigned to them in a similar way, by executing corresponding applications.



Site administration

Applications
Apps ✚Add ✎Change
Datas ✚Add ✎Change
Domains ✚Add ✎Change
Io registrys ✚Add ✎Change

Auth
Groups ✚Add ✎Change
Users ✚Add ✎Change

Relationships
Relationship statuses ✚Add ✎Change

Responsibilities
Assignments ✚Add ✎Change
Responsibilitys ✚Add ✎Change

Roleapp
Roles ✚Add ✎Change

Sites
Sites ✚Add ✎Change

Userprofiles
User Profiles ✚Add ✎Change

Add responsibility

Name: Secretary of Informatics and Telematics

Description: Secretary of Informatics and Telematics

Add assignment

Responsibility: Secretary of Informatics and Telematics ▾ ✚

Content type: user ▾

Object id: 12

Fig. 2. Responsibility creation and assignment.

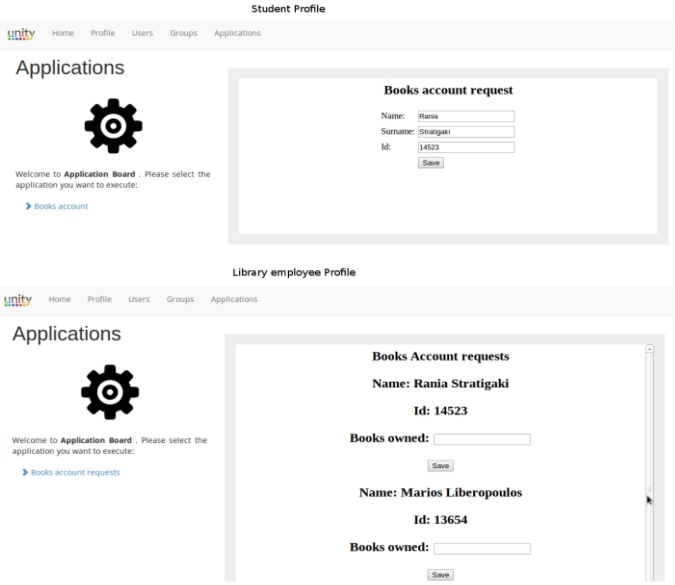
The issues arising in this case, handled by the extensions presented in this paper, are the following:

- each participant must only be allowed to execute the applications they are allowed to execute, based on their rights and responsibilities in the organization; in some cases, such responsibilities do not emerge from their role, as participants bearing the same role still may have different responsibilities.

- the participants are not be required to know the workflow and requirements in advance; in the real world case, the student would be informed of their tasks through the department secretariat, and they may have to advise the secretariat on many different occasions; in the social network case, the application composition mechanism is employed to indicate the composition of applications and therefore tasks.

Fig. 2 shows the responsibility creation and assignment screen for a specific responsibility, which handles cases when the first aforementioned issue occurs. Responsibility management is integrated within Unity administration environment.

Fig. 3 presents an indicative application execution example. All student participants may execute *Books Account Request* application, since the responsibility of executing this specific application is assigned to the student role, as shown in Fig. 2.



Student Profile

Applications

Welcome to **Application Board** . Please select the application you want to execute:

› Books account

**Books account request**

Name: Rania
Surname: Stratigaki
Id: 14523
Save

Library employee Profile

Applications

Welcome to **Application Board** . Please select the application you want to execute:

› Books account requests

**Books Account requests**

**Name: Rania Stratigaki**

**Id: 14523**

**Books owned:**

Save

**Name: Marios Liberopoulos**

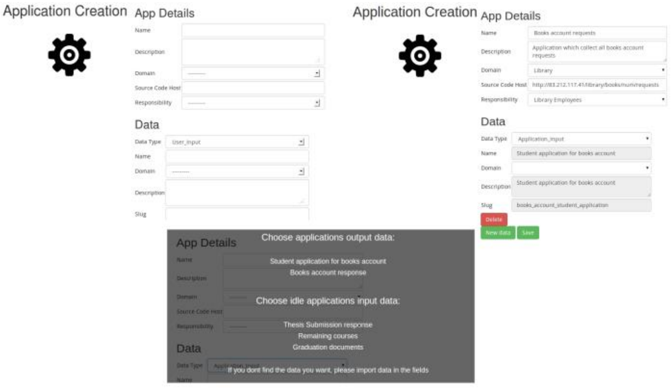
**Id: 13654**

**Books owned:**

Save

Fig. 3. Application execution based on responsibilities in the proposed extended interaction model.

Fig. 4 presents the application creation and registration page, which is part of the mechanism handling cases when the aforementioned second issue occurs. A developer in this case wishes to register an application named “Book account request”, specifying the URL of the source code and also the responsibilities for this specific application. Responsibilities, as mentioned before, might be assigned to certain roles or certain individuals (in this case Library Employees), and indicate execution rights. Additionally, the developer registers the application inputs and outputs; in case of inputs, they also have to indicate if they are user provided, profile data or expected from another application. In case the input data are provided by another application, the user is given the option to choose among existing applications outputs or indicate that they want to add a new data item; in the latter case, the input will be idle until the appropriate applications providing it are registered as well.


Application Creation
App Details
Name
Description
Domain ---------
Source Code Host
Responsibility ---------
Data
Data Type User_input
Name
Domain ---------
Description
Slug

App Details
Choose applications output data:
Student application for books account
Books account response
Choose idle applications input data:
Thesis Submission response
Remaining courses
Graduation documents
Data
If you dont find the data you want, please import data in the fields

Application Creation
App Details
Name: Books account requests
Description: Application which collect all books account requests
Domain: Library
Source Code Host: http://83.212.117.41/library/books/num/requests
Responsibility: Library Employees
Data
Data Type: Application_input
Name: Student application for books account
Domain:
Description: Student application for books account
Slug: books_account_student_application
Delete
New data
Save

Fig. 4. Application registration example.

Fig. 5 shows the graph created by the dependencies built by the applications registration procedure, as explained in the previous section. Dependencies are created at application registering time, when a developer indicates that the application being registered requires some data as inputs that are provided by other applications as outputs. This creates a dependency between those two applications, which can be seen as an arc in the graph. In case the application producing the input at hand is not yet registered in the social network, the dependency will be created if the developers confirm the suggested dependencies, when the all appropriate applications have eventually been registered.

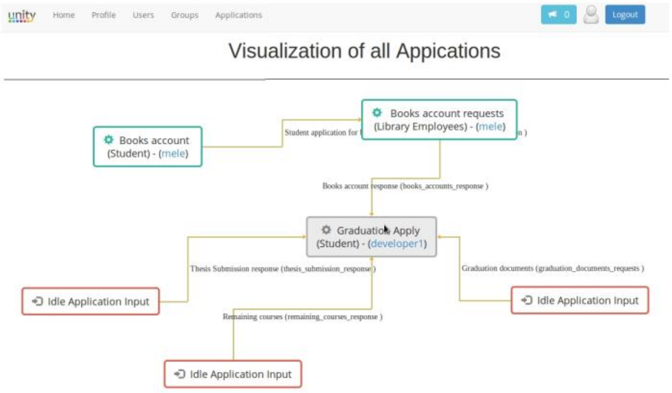

unity
Home
Profile
Users
Groups
Applications
0
Logout
Visualization of all Appications
Books account (Student) - (mele)
Books account requests (Library Employees) - (mele)
Student application for
n )
Books account response (books_accounts_response )
Graduation Apply (Student) - (developer1)
Thesis Submission response (thesis_submission_response )
Graduation documents (graduation_documents_requests )
Idle Application Input
Remaining courses (remaining_courses_response )
Idle Application Input
Idle Application Input

Fig. 5. Visualization of the applications dependencies graph for the data driven application composition mechanism.

## VI. Conclusions and Future Work

This paper presented extensions to the typical social network model to support and facilitate Social BPM, both to the role / relation interaction and collaboration model, as well as to the application execution mechanism of the Unity Framework. These extensions enable collaborative application execution and management, taking into account participant responsibilities in an organization, eventually leading to service provision through the social network, in contrast to merely information sharing or monitoring of task execution. In order to support the completion of business tasks over a social network, a set of features are proposed, providing enhanced functionality. More specifically, participants are engaged to perform specific activities by executing the corresponding applications, according to their role in the organization and current circumstances, and collaborate with others based on business policies and rules. The underlying social network framework provides ways to define such a role-based application execution mechanism, while it also accommodates for specific, more fine-grained responsibilities of certain participants. At the same time, it facilitates the management and discovery of cooperating applications in a uniform, generic and reusable way; the application management mechanism is data driven and as independent from specific applications as possible.

Future work in this area includes extensive experimentation with specific communities, organizations and enterprises, in order to apply and adapt the proposed model to specific requirements, and receive feedback on its adoption and use.

Currently, among the limitations of this research the issue of semantics is identified. Developers and administrators of the enterprise social networks constructed using the Unity platform are required to predetermine and work on common semantics for application data, in order to effectively achieve application interoperability. As a consequence, as a future step, we plan the inclusion of semantics in the reserved fields of the data driven application mechanism; such an extension would promote flexibility and enable the development of applications from more parties and external stakeholders, broadening the set of tasks that such a collaboration model can support.

Furthermore, for application data that are volatile and may be modified or expire with time, we plan to include an expiration date until which they are valid as application outputs, to be used as inputs for other applications. After the specific date, the specific application data will be registered in the user profile history and will not be able to be used in application compositions.

## References

[1] Activity Streams 2.0, 2013, http://activitystrea.ms

[2] A. Acquisti & R. Gross, 2006. “Imagined communities: Awareness, information sharing, and privacy on the Facebook.” In P. Golle & G. Danezis (Eds.), Proceedings of 6th Workshop on Privacy Enhancing Technologies (pp. 36-58). Cambridge, UK: Robinson College

[3] P. Anderson, 2007. “What is Web 2.0? Ideas, technologies and implications for education.” JISC Technology and Standards Watch.

[4] J. A. A. Bermejo, C. B. Bravo, M. J. R. Mateos, J. R. Piera, 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, pages 805-824.

[5] I. Bider, P. Johannesson and E. Perjons, 2011. “A Strategy for Merging Social Software with Business Process Support.” Business Process Management Workshops, Lecture Notes in Business Information Processing 66: 372-383

[6] M. K. Boulos, S. Wheeler, 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.

[7] I. Boughzala, 2012. :Collaboration 2.0 through the New Organization (2.0) Transformation." In I. Boughzala, & A. Dudezert (Eds.) Knowledge Management 2.0: Organizational Models and Enterprise Strategies (pp. 1-16). Hershey, PA: Information Science Reference.

[8] D. M. Boyd & N. N. Ellison, 2007. "Social network sites: Definition, history, and scholarship." Journal of Computer-Mediated Communication, 13(1), article 11.

[9] M. Brambilla, P. Fraternali, C. Karina and V. Ruiz, 2012. "Combining social web and BPM for improving enterprise performances: the BPM4People approach to social BPM." WWW (Companion Volume): 223-226

[10] G. Bruno, F. Dengler, B. Jennings, R. Khalaf, S. Nurcan, M. Prilla, M. Sarini, R. Schmidt and R. Silva, 2011. "Key challenges for enabling agile BPM with social software." Journal of Software Maintenance, 23(4): 297-326

[11] G. Bruno, 2012. "An Approach to Defining Social Processes Based on Social Networks." Handbook of Research on Business Social Networking: Organizational, Managerial, and Technological Dimensions, IGI, pages 272-286

[12] A. Dais, M. Nikolaidou, D. Anagnostopoulos, 2011. "OpenSocialGov: A Web 2.0 Environment for Governmental E-Service Delivery." EGOVIS 2011: 173-183

[13] J. DiMicco, D. R. Millen, W. Geyer, C. Dugan, B. Brownholtz and M. Muller, 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, pages 711-720

[14] F. Dengler, S. Lamparter, M. Hefke and A. Abecker, 2009. "Collaborative Process Development using Semantic MediaWiki." Proceedings of the 5th Conference of Professional Knowledge Management. Solothurn, Switzerland.

[15] W. Geyer, C. Dugan, J. DiMicco, D. R. Millen, B. Brownholtz and M. Muller, 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, pages 1545-1554.

[16] S. Gorg, R. Bergmann, M. Minor, S. Gessinger, S. Islam, 2012. "Collecting, reusing and executing private workflows on social network platforms." WWW (Companion Volume), 747-750.

[17] Grasso, A. Convertino, G, 2012. Collective Intelligence in Organizations: Tools and Studies. Computer Supported Cooperative Work (2012) 21:357–369.

[18] T. Gross, M. Koch, 2006. "Computer-Supported Cooperative Work - Concepts and Trends.", In Proc. 11th Conf. of the Association Information and Management (AIM), Bonner Koellen Verlag, LNI P-92, pages 165-172

[19] O. Hatzi, M. Nikolaidou, P. Katsivelis, V. Hudhra, D. Anagnostopoulos, "Using social network technology to provide e-administration services as collaborative tasks", International Conference on Electronic Government and the Information Systems Perspective and International Conference on Electronic Democracy (EGOVIS/EDEM), 2012.

[20] O. Hatzi, D. Topali, M. Nikolaidou, D. Anagnostopoulos, "Enabling Workflow Composition within a Social Network Environment", The 6th Workshop on Business Process Management and Social Software (BPMS2'13), in conjunction with the 11th International Conference on Business Process Management (BPM2013), Beijing, China, August 2013.

[21] S. R. Hiltz, 1998. "Collaborative Learning in Asynchronous Learning Networks: Building Learning Communities", 3rd WebNet World Conference of the WWW, Internet, and Intranet Proceedings, New Jersey Institute of Technology, pages 1-8.

[22] R. Hoegg, R. Martignoni, M. Meckel, K. Stanoevska-Slabeva, 2006. "Overview of business models for Web 2.0 communities", Proc. Workshop Gemeinschaften in Neuen Medien GeNeMe Dresden: TUDPress, Dresden, pages 33-49

[23] P. Johannesson, B. Andersson, P. Wohed, 2008. "Business Process Management with Social Software Systems - A New Paradigm for Work Organisation." 1st Workshop on Business Process Management and Social Software BPMS208, LNBIP V17, p 659-665.

[24] S. Kemsley, 2011. "Leveraging Social BPM for Enterprise Transformation." Book Chapter in Social BPM: Work, Planning and Collaboration under the Impact of Social Technology. BPM and Workflow Handbook Series, ed. Layna Fischer

[25] G. Kossinets, D. J. Watts, 2006. "Empirical Analysis of an Evolving Social Network", Science 6, Vol. 311 no. 5757, pages 88-90.

[26] R. Kumar, J. Novak, & A. Tomkins, 2006. "Structure and evolution of online social networks." Proceedings of 12th International Conference on Knowledge Discovery in Data Mining, New York ACM Press, pages 611-617.

[27] M. Leyer and N. Claus, 2013. "Toward an Agile Knowledge Connection of Employees with Regard to Business Processes." HICSS, pp. 3436-3445

[28] D. Lewis, 2006. "What is web 2.0?", ACM Crossroads 13, Vol. 13, Issue 1, pages 3-3

[29] H. Liu, P. Maes & G. Davenport, 2006. "Unraveling the taste fabric of social networks." International Journal on Semantic Web and Information Systems, 2 (1), IGI Global, pages 42-71

[30] C. McLoughlin, M. J. Lee, 2007. "Social software and participatory learning: Pedagogical choices with technology affordances in the Web 2.0 era", In Proceedings ASCILITE Singapore 2007, pages: 664-675.

[31] N. H. R. Motahari, C. Bartolini, S. Graupner, S. Spence, 2012. "Adaptive Case Management in the Social Enterprise." ICSOC, 550-557.

[32] G. Neumann and S. Erol, 2008. "From a Social Wiki to a Social Workflow System." Business Process Management Workshops, pp. 698-708

[33] OpenSocial Specification 2.5.1, 2013, http://opensocial.org

[34] T. Oreilly, 2007. "What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software." International Journal of Digital Economics No. 65, IGI, p 17-37

[35] B. Ploderer, S. Howard, P. Thomas, 2010. "Collaboration on Social Network Sites: Amateurs, Professionals and Celebrities", Computer Supported Cooperative Work (CSCW), Volume 19, Number 5, Springer, pages 419-455

[36] P. Pollara, & J. Zhu, 2011. "Social Networking and Education: Using Facebook as anEdusocial Space." In Proceedings of Society for Information Technology & Teacher Education InternationalConference 2011 (pp. 3330-3338). Chesapeake, VA: AACE.

[37] A. Richter and K. Riemer, 2009. "Corporate Social Networking Sites – Modes of Use and Appropriation through Co-Evolution." ACIS 2009 Proceedings, pages 722-732

[38] R. Schmidt and S. Nurcan, 2008. "BPM and Social Software." Business Process Management Workshops, 649-658

[39] N. S. Shami, M. J. Muller, D. R. Millen, 2011. "Browse and discover: social file sharing in the enterprise." CSCW '11 Proceedings of the ACM 2011 conference on Computer supported cooperative work, ACM New York, pages 295-304

[40] K. Swenson et al., 2011. "Social BPM: Work, Planning and Collaboration under the Impact of Social Technology." BPM and Workflow Handbook Series, ed. Layna Fischer

[41] J. Thompson, B. Doherty, 2006. "The diverse world of social enterprise: A collection of social enterprise stories." International Journal of Social Economics, Vol. 33 Iss: 5/6, Emerald Group Publishing, pages 361 – 375.

[42] Using LinkedIn Effectively, http://www.mindtools.com/pages/article/linkedin.htm.

[43] G. Vossen, S. Hagemann, 2007. "Unleashing Web 2.0: From concepts to creativity." Morgan Kaufmann.

[44] B. Williams, 2009. "Data Model for Social Networks", http://www.databaseanswers.org/data_models/social_network