Communities, Artifacts, Interaction and Contribution on the Web

Eleni Stroulia

Computing Science Department, University of Alberta, Edmonton, Canada
stroulia@ualberta.ca

Abstract. Today, most of us are members of multiple online communities, in the context of which we engage in a multitude of personal and professional activities. These communities are supported by different web-based platforms and enable different types of collaborative interactions. Through our experience with the development of and experimentation with three different such platforms in support of collaborative communities, we recognized a few core research problems relevant across all such tools, and we developed SociQL, a language, and a corresponding software framework, to study them.

Keywords: social networks, computer-supported collaboration, wikis, virtual worlds, web-based collaborative platforms.

1 Introduction and Background

The World Wide Web was conceived and born out of the desire to support information exchange, communication and collaboration. In its 30-year history (and it is flabbergasting to think about how short this history is in terms of time, and how dense it is in terms of events and innovations) it has more than fulfilled its promise and vision while at the same time undergoing three interesting transformations.

Originally, the objective of the web community was to enable document publishing and to advance large-scale information communication. The first beneficiaries of this platform were members of the academic and research community who had the knowledge and skills (a) to develop "web portals" even without any development tools, and (b) to access the published information through the original crude client applications. Through this activity, the first broadly usable clients and web-development toolkits were developed and gave rise to portals supported by traditional and new content owners, such as mainstream print publishers (MIT's Tech newspaper in 1991, BBC's TV program in 1994, and the Clinton White House in 1994) and new content providers (Yahoo in 1994). In this stage, the web was ***a web of information*** broadcasted by few to many.

The second phase in the Web's history was brought about by the advent of e-commerce sites (Amazon and eBay in 1995), which gave rise to ***the web of applications.*** In this new phase, the web became a ubiquitous platform, through which to deliver innovative services. The number of providers increased dramatically as the

M. Chignell et al. (Eds.): The Personal Web, LNCS 7855, pp. 131–150, 2013.

community became ever more creative about the types of services that could migrate to the web. The number of consumers also exploded with the increased availability of user-friendly browsers, search engines (Alta Vista, the first multilingual engine, was launched in 1995), and email-service providers for individuals (Hotmail was launched in 1996). Still, the communication model was that of broadcasting by relatively few to many.

This model changed with the advent of bulletin boards, originally associated with ecommerce web sites, and wikis and blogs, easy-to-use publication tools for individuals. These tools brought about ***the personal web***, a continuously available whiteboard, hosting everyman's opinions and personal expressions, across the world.

And as an increasing variety of tools has become available – for searching, tagging, visualizing and connecting personal posts, published through any of the multitude of available platforms – ***the social web*** has now emerged. Today each one of us is linked to a multitude of others through our on-line presence: to the authors of the blogs to which we comment, to the other consumers of the products and services we have bought or are considering buying, to the members of our professional communities (LinkedIn and ResearchGate), to the people whose micro-blog postings we follow (twitter), to our on-line friends (Facebook, Google+), to the members of our virtual-world communities (Second Life), and to the users of the on-line tools we use.

Clearly, the original web vision, of supporting collaboration, has been evolving throughout these phase transitions, and today, it appears that the potential for innovative modes of web-enabled collaboration has reached new heights. It is in fact at the core of the "smart planet"[1] ***interconnectedness*** vision, which includes (a) data, (b) system, and (c) people interconnectedness.

In our work, motivated primarily by the need to support collaborative software development, we have developed a family of web-based systems for supporting, managing and analyzing different types of collaborative activities. These tools have been motivated by different specific requirements, although they all belong in the general area of tools in support of web-based collaboration. In this paper, we aim to reflect upon this work as a whole and to place each activity within a common conceptual framework that could potentially drive further work in the field.

In the rest of this paper, we discuss the background of this work and we place it in the context of a two-dimensional design space, defined in terms of types of platforms on which collaborative tools are built and in terms of the collaborative interactions they may support (Section 2). Next, we review three tools developed by our group, all designed to support collaborative teams on the web (Section 3) and we discuss some interesting research questions pertinent to all these types of systems (Section 4). Next, we review our work on SociQL, a social query language designed to support different types of analyses across different social systems, such as the ones described in Section 3 (Section 4). Finally, we conclude with some thoughts on what we expect to be the next important innovations to come (Section 5).

[1] `http://www.ibm.com/smarterplanet/ca/en/overview/ideas/index.html?re=sph`

2 Collaboration in the Social Web

In the past several years, our team has developed three different web-based systems to support, manage and analyze a corresponding number of types of collaborative work. Looking back through this work, we have attempted to place it within a coherent conceptual framework by categorizing each tool in terms of two dimensions: (a) the types (and flexibility) of collaborative practices they support, and (b) the types of technology/platform on which they have been developed. Figure 1 illustrates this two-dimensional space and highlights a few interesting collaborative systems including our own.

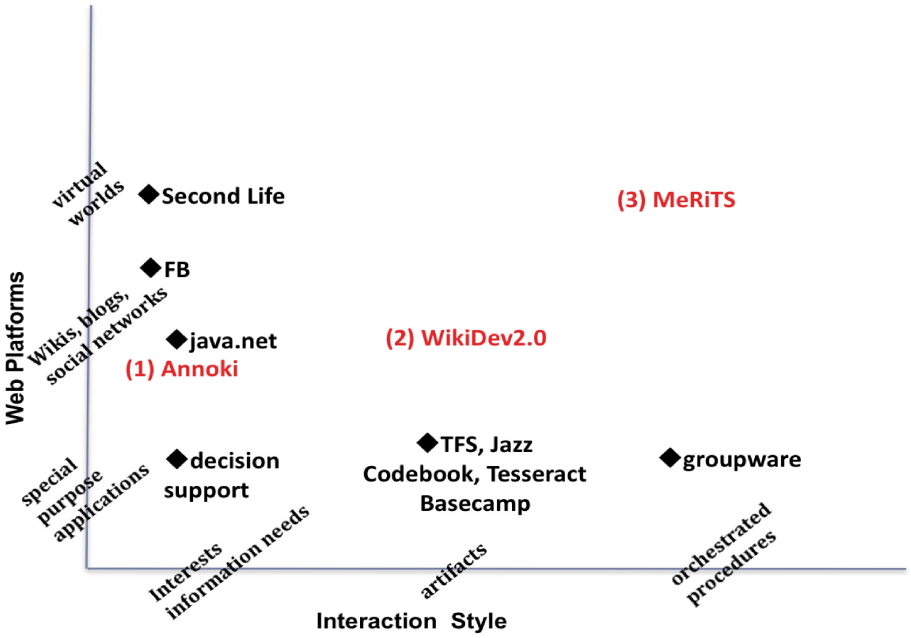

Fig. 1. Collaboration Tools in the Platform-Interaction Space

As shown in the vertical axis of Figure 1, the ***adopted platforms*** represent a spectrum of technologies**.** On one end of the spectrum, we have special-purpose applications, which are designed to support a specific task and make strong assumptions about the roles and capabilities of the collaborating individuals. This class of web-based applications exhibits a large variation in terms of the actual tasks they support, in terms of how strictly they regiment the roles and interactions of their members and the workflows of their activities, and in terms of their underlying architecture: they range from client-server systems with thick clients, to web-accessible applications with thin browser-based clients, to highly dynamic responsive Ajax-based clients. In this class, we generally understand the applications labeled as "groupware". In a still interesting – although not recent any more – overview of the

domain, Grudin [1,2] organizes the early groupware applications under the *workflow-support* and *decision-support* categories. The former category includes tools that aim to systematize the organization's tasks usually through form-based interfaces. The latter includes meeting-room and issue-tracking systems mediating decision-making activities among the members of an organization. The former tend to support well-defined activities whereas the latter are envisioned as supports for persons' decision making.

More recently, we have witnessed the emergence of a large family of traditional browser-based web applications that support the publication of content by individuals and the sharing of this content by more or less "formal" groups.[2] The overall intent in this category is not to directly support a purposeful collaborative activity of a well-defined group of people with distinct roles, but rather to enable the discovery of opportunities for content sharing and collaboration across the web, through *wikis*, *blogs* and *social networks*. Instead of task-specific roles, individuals are characterized by their own self-descriptions and by the reputations they build over time. And instead of workflow-orchestrated task-specific activities, individuals engage in general information seeking/publishing/sharing/editing behaviors.

Finally, the recent advent of *virtual worlds,* i.e., general-purpose massively multiplayer 3D environments, best exemplified by Second Life (www.secondife.com), has enabled a new breed of interactions mediated not by information but by virtual environments and objects, often – but not necessarily – simulating real spaces and real objects. Some of these environments focus on a particular activity, such as playing games or listening to music, but many of them are designed to provide a virtual environment, in many ways parallel to the real world; for an overview of the available virtual worlds circa 2009, the interested reader should refer to [3]. In these environments, the general objective is not simply to share information but to express oneself and meet other like-minded virtual-world denizens. In addition, in these environments, one can engage in simulations of real-world activities for the purpose of entertainment (i.e., dancing), commerce (i.e., buying and selling virtual goods and services) and education (i.e., simulating professional procedures). Virtual worlds are increasingly being used for supporting a wider variety of applications; beyond providing a virtual environment, they are used to augment with an alternate layer applications and services in the real world. In our own work, we have explored three different types of virtual-world uses as mirrors or the real world, or as alternative places in which to conduct realistic simulations relying on real-world systems, or as trans-reality places [4,5].

In the horizontal axis of Figure 1, we have identified three interesting points in the spectrum of collaborative practices, identified in terms of the "contract" that the collaborators assume about "what is being shared". At the very ***informal*** end of collaboration practices, we have fluid communities of individuals with common interests and/or expertise. Participation in these communities requires only that

[2] For a comprehensive list of social applications today, the interested reader should visit `http://www.theconversationprism.com/` for an interesting infographic of the social media universe today.

individuals share a common natural language, in terms of which to express themselves and contribute to the community conversations. For example, regular blog readers interact through their blog entries and Second-Life users interact with each other through voice and text messaging.

As the degree of the community formality increases, participation requires a shared common understanding of what artifacts are shared within the community. For example, participants of a retailer's loyalty program may share product reviews, structured in terms of particular questions of interest, or buyers/sellers of pre-owned goods may describe their products in terms of standard conventions, including pictures, description, original cost, and level of use. At this level, the objective is still information sharing, although this information may directly support an activity; for instance, information about an object on a web-based marketplace is intended to support an economic transaction, which, however, may not be mediated by the marketplace web application (the two parties may simply contact each other through email).

As the community matures further, a shared understanding emerges about the activities that can effectively be supported based on the information shared by the community members. As an example, consider software teams using web-based software-productivity tools, such as version-control and project-management tools. These tools imply a specific set of activities, i.e., adding, committing, updating files; editing tasks and timelines; and keeping track of one's own working hours. These activities are supported directly by the collaborative web application, although they are not necessarily strictly orchestrated. Similarly, when visiting a virtual-world clothes' retail space, users may try clothes on their avatars, decide to buy them and pay for them with virtual currency. The virtual-world does not orchestrate these activities, but the owners and visitors participate in these transactions guided by their shared understanding of how retail works.

The tools described in the next Section are examples of these three types of collaborative practices, developed on two of the most recent commodity social platforms, namely wikis and virtual worlds. Our experience with these systems led to the formulation of the research questions in Section 4, and has motivated the design of SociQL (described in Section 5) as a language and an implementation framework to study these questions. Section 6 concludes with a summary of our experience to date and outlines some promising avenues for future work.

3 A Suite of Collaborative Platforms

Our team has been exploring this space of collaborative web-based tools and has designed and developed a suite of systems across this space, which we describe in some detail in the remainder of this Section. For each of these tools, we first discuss the requirements motivating its design, next we review its major functionalities, and finally, we point out the most interesting findings through our experience with it. Our experience building and evaluating these tools eventually led us to recognize some common themes in the nature of the data collected in these tools, the interactions of

the tools' users, and the analyses that might be of interest to the members of the collaborative teams and their managers. We explore these themes in Section 4. More recently, we started exploring these common themes in the design and development of SociQL, a language for querying and analyzing systems supporting the activities of teams, and, more generally, social networks. SociQL is described at a high level in Section 5, and in more detail in [6].

3.1 Annoki

Annoki [7] is a wiki-based collaborative platform, designed to support communities of interest, sharing information based on their members' common interests. The original envisioned usage scenario for Annoki was to support the collaboration activities of our research team, such as keeping track of the lab software and hardware resources, maintaining a list of the publications relevant to our research and our discussions about them, and maintaining a research log including research problems, methodological decisions, development plans and timelines and ideas explored, rejected, modified or under consideration. Having such a shared resource recording our research activities was intended to support the authoring of theses and publications and to enable the faster on-boarding of new team members.

We chose to develop Annoki on top of MediaWiki primarily because of the general popularity of the MediaWiki platform, which implies an easy learning curve, as most users are somehow familiar with it, if only based on Wikipedia, the most popular MediaWiki tool. Second, and more importantly, MediaWiki provides a natural metaphor for understanding the general class of communities-of-practice applications, where information resources (corresponding to wiki *pages*) are shared and collaboratively manipulated by *users*. Users can informally and collaboratively edit the shared resources, and can reflect upon them and discuss about them on special-purpose *discussion pages* attached to each page. The evolution of the resources, the users, and their communications is automatically tracked and reported through the *recent-changes* and *history* MediaWiki tools, which can also be configured to push email notifications to the users interested in *watching* particular pages of interest. MediaWiki, as is the case with wikis in general, does not yet afford rich data views, but its software architecture is extendible enough to allow us to alleviate this shortcoming. The Annoki toolkit adds the following set of extensions to the MediaWiki features.

***Namespace-Based Access Control*:** Wikis are designed for open sharing and collaborative editing of resources, providing only very coarse-grained access-control mechanisms: a page can either be editable by community members or the general public, or not. However, in the context of organized groups, individuals usually assume different roles, with different privileges, which could (and frequently should) be recognized and/or enforced by their collaboration tools.

In Annoki, we piggybacked a flexible access-control system on the MediaWiki concept of *namespaces*. A namespace is a cluster of related pages, the title of which start with the same prefix, i.e., the namespace title. Each Annoki user has his own

namespace and, by default, all pages he creates belong in this namespace. Additional namespaces can also be defined to organize wiki pages that belong to a group of users. A namespace can be "public", in which case everyone can read its pages, and if they are not protected, edit its pages. If a namespace is not public only the users associated with it can read/edit its pages. Users can be flexibly assigned to multiple namespaces and pages can move from one namespace to another to increase/restrict their visibility. In this manner, layers of access rights can be supported for personal, project-specific, organization-related, and publicly accessible content.

***Interactive Visualizations*:** Wikis were originally conceived as hypertext platform, where users share textual content, structure its presentation through a limited set of formatting metadata, and link it to existing content through hyperlinks. Although each page editor may have an understanding of how his pages connect to the rest of the wiki content, find one's way through the links presents a challenge to new visitors. The wealth of information that is accrued in a wiki comes at a cost: the more pages a Wiki has, the more difficult it becomes to navigate through it. It is common for a Wiki page to contain a sizable number of links in its body text. Users have to read through the text to locate these links and to guess where to navigate next. It is also easy for users to feel "lost" after having gone through a number of links, since the standard wiki interface does not provide any navigation context. In our earlier work [8] we studied how a graph-based representation of a wiki content, where individual pages are represented as nodes and URLs between them are represented as edges, impacts the ability of users to search through the wiki content and answer pertinent questions. In this study, we found that searching through such a graph indeed decreases the number of pages that a user has to visit to complete the task.

Inspired from this study, Annoki is equipped with a rich, interactive, Ajax-based visualization, WikiMap, a visualization of the whole wiki structure (users, pages, links among pages and authorship relations between users and pages). WikiMaps enable Annoki users to obtain a global view of its content and its evolution history. In addition, wiEGOs (wiki-integrated Electronic Graphic Organizers [9]) are visualizations of the semantic structures implicit in a set of special template-based pages (i.e., tree, topic, persuasion, brainstorm, story, and decision maps, as well as flowcharts). wiEGOs extend the template concept of MediaWiki, by associating a visual structure with the standard organization structure of pages that are instances of templates, enabling the creation and editing of template instances through a visual interface.

***Contribution Analysis*:** Wikis are designed to support collaborative content development, without focusing on individual contribution: looking at a wiki page, one cannot distinguish the individual contributions of its editors. As wikis are adopted for institutional purposes, where contributors are likely driven by career goals, it becomes interesting to support the ability to recognize each editor's contributions. In [10] we studied the question of how to assess the editors' contributions by developing a suite of sentence-based metrics and comparing them against users' perception of contribution. Through this study we found that users' perception of contribution was

strongly correlated with significant content addition and deletion, as well as hyperlink addition.

Driven by our original hypothesis, that in institutional contexts wiki users are (at least partially) driven by their desire to distinguish themselves among their peers through their contributions, we developed a special purpose pie-chart visualization to communicate the type and percentage of each individual editor's contribution on a page. This visualization can clearly convey the various contribution metrics to the wiki users. Through interviews with potential users, we concluded that such tools are likely to change wiki editors' behavior, motivating editors to contribute in ways that would result in increasing their "slice" of the pie chart, thus making their contribution obvious to the wiki community. Through our experience with Annoki after this visualization was actually deployed, we collected anecdotal evidence that this is indeed true, namely explicit communication of contribution motivates further contribution, at least to some editors; however, we have not conducted a formal experiment to precisely investigate this phenomenon.

Experience with Annoki: Through our experience using Annoki, we discovered that the biggest challenge in the adoption of the tool was the lack of a well-defined process guiding its use. The most satisfied users were those who adopted (or developed on their own) a consistent style for editing, structuring and linking their pages, such as daily task records, for example. Without such a consistent usage style, searching and finding information becomes more difficult, thus substantially undermining the usefulness of the tool. Support mechanisms such as visual exploration of the wiki content (through the WikiMap) and additional page meta-data (such as keywords) can help users to more easily find information [8], but they are effective when they reflect a conceptual organization structure underlying the wiki content.

3.2 WikiDev2.0

Having used Annoki for over three years in our lab, we recognized an opportunity to adopt it as a lightweight tool for supporting the collaboration among software developers working on team projects. Traditional software-collaboration tools, such as version-control repositories and bug trackers, have focused primarily on supporting the sharing and management of technical software artifacts, such as software source code, other assets like configuration files and scripts and images, formal documentation and tasks. However rich information can be extracted by understanding the relations between activities across tools and by analyzing the informal communication among developers, in terms of emails and text messages. Recognition of this fact has given rise to the software-engineering field of "mining software repositories" and was the motivating factor behind our WikiDev2.0 [11,12,13,14] project.

WikiDev2.0 was conceived as a lightweight platform through which (a) to integrate information about the software artifacts produced in the variety of tools used by the software team (code, documentation, communication messages etc), (b) to analyze this information in order to infer interesting relations among these artifacts,

the team members and their activities, and (c) to present views on this information and its analyses that cut across the individual tool boundaries. In WikiDev2.0, a software team is assigned a namespace and information from the different tools the team is ingested in different types of template-based wiki pages. To date we have integrated SVN, Bugzilla, e-mail, and IRC with WikiDev2.0. Each file in SVN is represented as a wiki page and its history can be explored through wiki differencing. Bugzilla tickets are also represented as wiki pages. Each team is associated with a mailing list, whose archive is ingested as yet another wiki page, and an IRC channel, monitored by a chat-bot that records the exchanges through this channel in another wiki page. In addition to these automatically constructed pages, team members can create wiki pages for whatever purpose they choose. Throughout our experience over about four years using this tool to support undergraduate software-engineering courses, we noticed that teams created pages to keep track of their internal timelines, their weekly progress and their own information documentation of different aspects of the software project. Finally, each project namespace has a page that includes visualizations of several straight-forward productivity and communication analyses, including how frequently team members commit in SVN, how frequently changes to the project files are committed, and how frequently each team member communicates with the others.

WikiDev2.0 caters to the needs of small communities – i.e., software teams – engaged in a purposeful activity – i.e., the development of software – producing structured artifacts – i.e., source code, test suites, and documentation. This is different from the usage scenario envisioned for Annoki, which was conceived to enable a researcher community to share information of common interest, where the interests were loosely and implicitly defined by virtue of membership in the community. In WikiDev2.0 we revisited and reformulated the question of contribution analysis: instead of developing metrics to evaluate the contribution of individuals to different (types of) pages, we focused instead on recognizing the dependencies between contributions of different types.

The ***code and communication clustering*** analysis of WikiDev2.0 [11] was conceived as a means of correlating technical software artifacts and their contributors with contributions to informal communications (i.e., email messages and IRC chats). The analysis consists of the following steps. The first step involves parsing of all the textual information associated with the input information feeds, to recognize mentions of team members (their names, nicknames, or IDs) and software artifacts (classes, methods and interfaces). The recognized references reflect the *explicit relations* between people, code, and communication artifacts. A subsequent step calculates further *implicit relations* and provides further insights about hidden dependencies among these artifacts. This analysis is performed within non-overlapping, sliding, week-long windows, and the result is the correlation of developers' contributions to software artifacts with communications among the developers. Examination of the resulting clusters led us to recognize discussions among the team members about specific artifacts and who is (should be) currently working on what artifact. Such discussions can be potentially relevant in identifying people who should be consulted when a particular artifact is being maintained, thus actually informing the team members' work.

In the above process, relations among the various data elements contained in the wiki are inferred on the basis of shared keywords, i.e., shared team-members' names and class/method identifiers. Aiming to better analyze the WikiDev2.0 textual content, we developed a ***syntactic-semantic text-analysis*** method [14]. The method consists of three steps. First, all sentences from WikiDev2.0 wiki pages, email messages and IRC chats are extracted and parsed. The resulting parse trees are annotated with semantic tags, based on a (partially project specific) vocabulary of known terms. For example, all references to programming languages, tools and activities (such as developing, testing etc) are recognized and correspondingly annotated, as are all mentions of team-members' names and code-artifact identifiers. Finally, a set of rules, such as "*S:<team-member name> V:develop O:<class-identifier>*", are matched against the annotated syntax trees to extract subject-predicate-object triples, corresponding to relations such as "*who worked on what*", "*who has experience in what*" etc.

Both the above analyses were conceived as a means to understand the software team process. To help the team members themselves improve their awareness of their project artifacts and activities, we developed two alternative visualizations [12] of the WikiDev2.0 content. The first was a traditional graph-based visualization accessible through WikiDev2.0 itself; the second was developed based on a city metaphor in the OpenWonderland virtual world. We experimentally evaluated the relative MERiTS of the two visualizations by having subjects access them to answer questions about the underlying software project. We found that the two views were rather complementary: in most questions where the subjects had trouble finding the answer in one system, it was easier for them in the other system. However, overall the more traditional WikiDev2.0 graph-based visualizations seemed to be a slightly easier platform. This can be attributed to the fact that WikiDev2.0 has a more intuitive and familiar interface (hyperlinks and web pages) compared to navigating in a virtual world. However, neither platform showed a clear advantage over the other since half of our small set of subjects (4) preferred WikiDev2.0 while the other half chose WikiDev3D.

Experience with WikiDev2.0: Through our experience using WikiDev2.0 we found that the feature most appreciated by the developers was the feedback they received by the instructor and TAs on their deliverables. Using the team's WikiDev2.0 area as the repository of all team deliverables and feedback enabled the tighter interaction and gave the teams more confidence about their progress. We found that the clustering analyses we developed on the WikiDev2.0 content, whether through simple information-retrieval mechanisms [11] or through natural-language analyses [13], did provide useful insights to the instructor about the team interactions and progress, which could potentially enhance the instructor's ability to provide useful feedback. With respect to supporting the developers' awareness of their project, there is a lot of work in the area exploring various styles of information visualizations and dashboards. Many of these mechanisms are promising although no conclusive overarching theory has emerged yet on the developers' information-access patterns of their software repositories.

3.3 MERiTS

The MERiTS system [15,3,16,17,18,19] is the third of our collaborative-work support tools and it focuses on collaborative activities that are more complex and involve complex orchestrated interactions among individuals, in addition to the exchange of textual information (as in Annoki) and sharing of information about well-defined work products (as in WikiDev2.0). Most professional roles require knowledge and procedural skills in performing role-specific tasks as well as communication skills for interacting with other professionals, and it is increasingly recognized that the role of education is to enable students to competently perform in their eventual professional roles, as opposed to just acquiring facts and learning to perform procedures. In this light, traditional textbook-and-lecture teaching methods are gradually giving way to simulation-based case scenarios, where students have the opportunity to experience situations in which they have to apply their knowledge and skills as would be expected of them in their future professional roles. However, conducting simulations is a challenging proposition, given the costs associated with acquiring the necessary equipment and hiring instructors and facilitators to conduct the simulations, and the limited number of students that can participate in any given simulation session. Virtual worlds are online platforms that combine the accessibility and collaboration possibilities of the web with the immersive, realistic qualities of virtual reality, offering an appealing and cost-effective alternative for conducting simulations in the context of competency-based training programs. Our MERiTS tool [19] was conceived to enable (a) instructors to specify educational scenarios, and (b) students to experience those scenarios in a realistic, interactive manner.

The MERITS framework offers two important features. The first involves a method and tool support for ***specifying complex collaborative processes,*** in terms of BPEL workflows, including tools for specifying the behavioral capabilities of the various roles in the process in terms of web services invoked by avatar actions in the virtual world, as well as developing behavioral scripts for real-object simulacra in the virtual world. The second important feature is a ***comprehensive action-recording*** tool [18] that produces a compact synchronized trace of all in-world actions of all simulation participants, which can then be parsed to identify action patterns of pedagogical interest.

The MERITS architecture mimics the three-tiered structure of traditional web-based applications, with a virtual world as the user interface, a BPEL orchestrated set of software services as the application logic (that is, the software implementing the automated activities of the service-delivery process), and a resource repository maintaining a record of the archival data of the organization and the transient data of each service-delivery simulation scenario. Instructors can specify relevant educational entities by updating the resource repository through web-based forms, accessing REST APIs of the repository. The BPEL workflows that specify the behaviors of people and objects in the scenario may, in principle, be created using graphical, web-based tools. However, there are conceptual challenges involved in the specification of a BPEL workflow that make merely providing a graphical interface insufficient for removing implementation barriers for non-technical users.

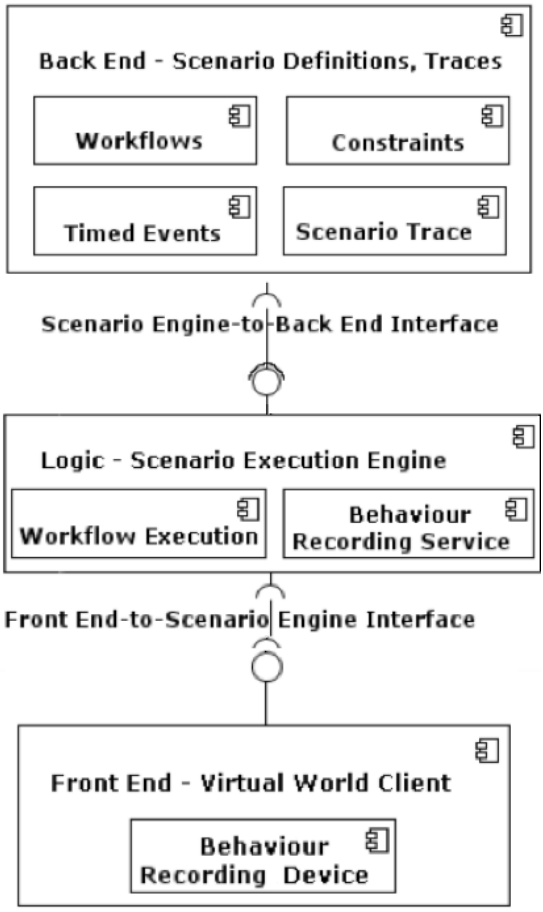

Fig. 2. The MERiTS Software Architecture

At run time, the BPEL workflows are enacted through the interactions of people and objects in the virtual world and through the behaviors of underlying automated software systems. When a student performs an action through his or her avatar, a behavior script is executed in the virtual world. The execution of this script may (a) change the state of the virtual world and (b) change the state of the corresponding workflow, shown in the second tier in the diagram in Figure 2. In our implementation, the server interprets the action in the context of the overall process workflow to determine how the scenario should proceed in response to the action. The BPEL workflow can also be connected to external devices, thus allowing the simulation to extend beyond the boundaries of a particular educational institution. For example, in a healthcare education context, where we have primarily evaluated MERiTS, the system may be connected to a web service that provides simulated patient data.

The MERiTS system enables the scalability of simulation-based teaching and learning. On one hand, virtual-world simulations can be cost-effectively accessible by

many more students, across geographical and institutional boundaries, than their more realistic counterparts. On the other, the trace-recording and analysis functionality is essential for "scaling up" the capacity of instructors to evaluate the competence of their students as they go through simulations; by inspecting the recorded trace and its analyses, instructors can obtain a good understanding of the their students' competence.

The question of relative contribution analysis for each of the participants is still relevant but takes a different form in MERiTS: a person's contribution to the accomplishment of the simulated activity is measured by analyzing the recorded simulation trace to calculate the distinct activity steps taken by this individual, the correctness of the timing of these steps, and the lack of violations of the activity constraints.

Experience with MERiTS: We have deployed MERiTS in several different scenarios, which can be classified in one of two categories. In the first case [4], the scenario simulates a classroom setting, replacing a more traditional web-based teleconferencing tool. In a study where students in an interprofessional health-sciences course met and planned the discharge of standardized patient in the virtual world, we faced interesting usability challenges since many participants found it difficult to learn how to interact in the virtual world. Moreover, some facilitators "lost control" of their classrooms as inattentive students starting experimenting with the tool distracting everyone from the educational activity. In this deployment style, we found that the fairly large number of students in the classroom made it difficult for the students and the instructor to migrate their normal class-interaction patterns (being attentive, taking turns, etc) to the virtual world.

In a later case study of individual nursing students simulating an asthma-emergency scenario latter type of scenarios [19] we found that the students had much less difficulty interacting with the virtual world, even though the environment was more complex. In this study, students were more prepared (they went through a tutorial before class) and found the experience "low pressure", "less threatening than their clinical hours" and "valuable for nursing programs". In spite of the students' overall positive experience with the simulation, our analysis of their competence through pre- and post-simulation questionnaires did not show improvement that could be attributed to learning through the simulation; this finding is in agreement with much current education literature.

4 Common Themes in Social Applications

Developing and reflecting upon the three systems we described in the last section, we have come up with a set of research questions (and associated technical challenges) that cut across most, if not all, web-based social systems today. We review these questions in this section, organized in two different groupings of "analysis questions" around social systems and possible "services supported" by social systems.

Today, there exists a plethora of social-networking sites, each one supporting different types of "connections" among members and catering to different

demographics. Some sites enable bi-directional connections, like Facebook; others enable the organization of connections in conceptual clusters, like Google+ circles; yet others only support directed connections, like Twitter. MySpace caters to a younger demographic than Facebook, which in turn is surpassed in popularity by Orkut in Brazil. In addition to these "superficial" differences, each of these social networks encourages different types of communications. Facebook appeals to people who want to keep in touch with family and friends, where Twitter seems to be the medium of choice for people who want to share and access information from a wide variety of channels. Facebook favors deeper connections and enables the organization of these connections in groups so that different personas can be projected to each of them. Google+ takes this idea even further enforcing people to register their true identity with the system but supporting the differentiation of the spheres of socialization through circles. Twitter, on the other hand, encourages maximization of connections (followers) and enforces a single persona on its users, who cannot distinguish their followers in groups.

Clearly these differences deserve deeper analysis; in the mean time, all of these networks share three important concepts, i.e., they define and support ***communities***, that enable ***contribution*** through sharing of different types of content over different types of channels, and as a result they enable individuals to ***influence*** each other.

4.1 Recognizing Communities

Groups of collaborating people are not uniformly cohesive. Some members are more highly connected to each other than to the rest of the group. This is a corollary to the "homophily" [20] phenomenon. Homophily is the tendency of individuals to associate and bond with similar others. Individuals in homophilic relationships share common characteristics (beliefs, values, behaviors, etc.) that make communication and relationship formation easier. If homophily is indeed a pervasive phenomenon in groups, the question becomes (a) how one might recognize the relations that underlie it in each particular collaborative/social system, and (b) how one would go about supporting the emerging homophilic groups (i.e., sub-communities).

Let us review the issue of "recognizing and supporting sub-communities". It is a problem relevant to all the three systems we discussed above, but its various instances differ from each other and, therefore, the three systems address it in slightly different ways. Annoki members are associated with individual- and project-related namespaces that contain pages; the implication is that sub-communities are either person- or project-centric. In Annoki, there is a single relationship among users, namely page co-editing, therefore the namespaces completely capture the relations among users: users co-edit the pages in their namespaces. As a result, there is no need to recognize implicit sub-communities. Individuals are explicitly assigned to projects, or, alternatively, based on their evolving interests, they may search and find interesting content and decide to join (and leave) the projects in which this content belongs.

WikiDev2.0 also supports the organization of individuals in projects. Individuals do not flexibly choose their groups; they are assigned to them and the user-project

relationship does not change. However, within a project team, there are many possible relations among the team members, such as working on the same artifacts, working on the same tasks (i.e., tickets), communicating with each other through one channel (email) or another (wiki pages). We have analyzed these relationships of team members, both direct (through common artifacts) and indirect (through communications), to recognize subgroups of individuals who have frequently communicated with each other. Furthermore, through further inspection of the resulting clusters, we can examine how frequently two team members belong in the same cluster. Such co-occurrence of members in clusters would implicitly indicate an increased degree of collaboration of the corresponding team members, which may or may not be reflected in the explicit communications of these team members in email.

In the workflow-defined simulations of MERITS, the activities and the relations of the participants are explicitly represented in terms of the workflows they enact. However, in cases of more open-ended activities within a virtual classroom, special-purpose relations (like communication) can be used as the basis for recognizing sub-communities of people who interact more closely with each other.

4.2 Recognizing Contribution

As the collaborating community increases, the nature of what is being shared becomes less well defined, and the nature and amount of an individual's contribution becomes more difficult to discern. MERiTS workflows usually involve small teams of about 3-4 professionals that collaborate in well-defined ways, interacting with task-specific (virtual) artifacts and orchestrating their interactions in terms of BPEL-specified workflows. In WikiDev2.0, most teams consist of four to six developers (plus TAs and instructors) and the team members share software artifacts (which, however, are produced using tools external to WikiDev2.0). In contrast, the Annoki installation in our group has about 200 members (some of whom are not active any more) who communicate through natural-language text.

MediaWiki, as well as most wikis, offers a differencing capability, which summarizes the contribution of an individual to a specific version. Annoki provides a more sophisticated contribution analysis and visualization tool, which summarizes the contribution of an individual to a wiki page over its lifecycle [10]. WikiDev2.0 implicitly recognizes contribution through its visualizations of the frequency of SVN commits, wiki-page edits, and email communications. Furthermore, analysis of the clusters in WikiDev2.0 can shed further insight on the contributions of an individual, although this is not an automated inference. MERITS measures contribution in terms of actions taken in the context of accomplishing a collaborative workflow. Through its recorded activity logs one can define further metrics of interest based on the participants' in-world activities and measure contribution in different ways. For example, one can imagine that it would be interesting to identify the persons who talked the most during a session or the person who made the most interactive gestures (like shaking hands for example) with others.

This discussion is motivated by the assumption that a person's "importance" within a community is related to the person's "contribution" to the community's activities

and assets. Many domain-specific importance metrics can be based on different person attributes, but contribution appears to be a cross-domain metric of an individual's importance within the community.

Related to the concept of contribution is the concept of influence. Within a collaborating community, people influence their collaborators through their contributions. Not all contributions however are equally likely to be "consumed" by other team members and to influence other people's contributions. In WikiDev2.0, the clustering process implicitly attempts to recognize the members' influence to code artifacts by collecting references of other materials, associated with team members, to these artifacts. In MERiTS, influence can be perceived directly through analysis of the workflows, where data/artifact production/consumption relationships as well as ordering relations between steps, can be assumed to define influence of the producer to the consumer and of the preceding to the subsequent actor.

Recognizing densely connected clusters of individuals, based on their shared properties, their relations and the strength of these relations, and understanding the interactions among individuals and the flow of influence that these interactions imply are two important recurring themes in social/collaborative systems. These phenomena are of interest to sociologists and, at the same time, understanding how they manifest themselves in a particular community is key to more effectively supporting the community and its objectives. SociQL was designed to support the representation of questions relevant to these themes and the access of social systems to answer these questions.

5 SociQL

Through the above discussion on the collaborative tools developed by our group (Section 3) and the two interesting (and recurring) problems in the general area of social platforms (Section 4), we have explored some of the variety in the area of social/collaborative applications. Nevertheless, research, aiming to understand how these communities work and how they can most effectively be supported, is fundamentally interested in answering a common set of core questions. These questions include, but are not limited to, the following: What types of individuals belong in the community? How are they related to each other? How do these relations become evident? What are the interesting substructures through which subgroups of community members are related? Who are the most influential individuals in the community? How does the community evolve over time and space?

The above questions are fundamentally sociological in nature, and, in order to be able to systematically explore them across a variety of applications, a *social query language* is needed. The syntax of this language should express the concepts of individuals, relations, communities, structure and influence with first-order primitives and should support the intuitive expression of the above questions. Furthermore, this language should be associated with a systematic methodology for how it should be mapped to the concepts of each particular social-collaborative application.

SociQL [6,21] is a language developed by our group to meet exactly the above requirements. Unlike generic web-query languages, SociQL is designed to support the

examination of sociological questions, relying on the *object-centered sociality* theory [22]. While recognizing the social interaction between *individuals*, this theory exalts the role of *specific objects* as the reasons why social actors affiliate with each other; essentially the theory assumes that objects constitute the reasons why (and the evidence of) people relating to each other in communities.

For this reason, we define the SociQL data model around the concept of an ***object***. For instance, in the context of WikiDEv2.0, a class (an object) connects the team members who have contributed to its development. Similarly, a mail message (an object) connects the parties (sender and recipients) who have access to it. In the MediaWiki-based Annoki, the wiki pages are the objects that connect the pages authors. In MERiTS, the avatars are connected through the simulacra of the real-world objects they manipulate as well as through their communication objects, i.e., their text and voice utterances. Each SociQL object is represented in terms of a pair, associating a unique identifier with the object type. ***Relations*** among objects are assumed to be binary and are represented as tuples, consisting of the identifiers of two related objects and the type of the relation.

In SociQL, both objects and relationships are described by ***properties*** (actual data), such as the login ID of a team member (object property) or the timestamp when a developer last committed a class (relation property). The properties of an object are represented as tuples associating the object identifier with a property and its value. The properties of a relation are represented as tuples associating the participating object identifiers, the relation type, the relevant property and its value.

SociQL also distinguishes the ***context*** in which properties are defined to describe the objects and relationships. For instance, the same query might return different email addresses for the same individual depending on the context in which the query is asked[3]: in Annoki, the email address of the individual as a researcher will be returned, while in WikiDev2.0, his address as a student participating in a course will be returned. In practice, each context corresponds to a different social/collaborative system. It is the notion of context that enables us to use SociQL to interlink different community platforms and integrate the knowledge about individuals and their relations that is currently hoarded in a variety of silos.

Once the objects, relations and properties of a social application have been mapped in SociQL, i.e., SQL queries or REST APIs have been implemented to retrieve them from the subject system and expose them in the simple SociQL representation described above, one can examine several interesting sociological questions relying on the algorithms implemented the SociQL engine.

First, one can express basic questions in terms of basic SociQL queries as follows:

SELECT *relation-label*$_1$(o_1, o_2)
FROM object-type o_1, object-type o_2, *relation-label*$_2$(o_1, o_2)
WHERE o_1.*property-label=property-value*.

[3] Note that SociQL has not been integrated with Annoki, WikiDev2.0 or MERiTS; it has been integrated with two different systems (see [20]). Our examples in this section on its use with Annoki, WikiDev2.0 and MERiTS are envisioning its application in these systems.

The above query will first identify all object pairs (o_1, o_2), where the value of the property *property-label* of o_1 is *property-value,* and o_1 and o_2 are related through the relation *relation-label*$_2$. Next, it will select the subset of these pairs that are also related through the relation *relation-label*$_1$.

More interestingly, building on the above basic query, one can retrieve dependency ***paths*** between objects. A path expression implies a sequence of objects o_0, r_1, … o_i, r_{i+1}, ..., r_n, o_n, where r_1, ..., r_n represent relations, o_0, ..., o_n denote objects, and for each i, there is a relation r_i between o_{i-1} and o_i. Starting from the object o_0 and ending in the object o_n there may be several paths that match this expression. Path expressions are helpful to find all connections and possible influence zones in social networks. For instance, when querying for all paths connecting two developers in WikiDev2.0, we might discover that they are related through their co-editing of the same class, or through their exchange of an email message, or through the fact that one developed a class depending on a second class developed by the other.

In addition, the SociQL toolkit includes the implementation of several centrality measures that can be used to filter the results returned by any SociQL query. In this manner, after having discovered all the developers with whom a particular developer is connected through paths of up to n steps, we may filter (and or sort) them according to their importance in terms of their centrality according the email-sharing relation, in order to find who of these retrieved connections is the most prolific email correspondent.

6 Summary and Conclusions

In this paper, we have attempted to review, in broad strokes and through the perspective of our own experience, the general area of social collaborative platforms. We believe that this area of research activity and real-world-driven innovation has revolutionized once more our idea of the role of the web. From a repository of authoritative information, to a distributed application platform, to a forum of personal expression, to a community, the web has now become fundamentally entrenched in our every-day activities. And, given the variety of activities currently taking place "on the web" in some form or another, and the variety of web-based platforms that one can adopt to develop tools to support these activities, developing, managing and analyzing these collaborative web-based systems is a unique challenge.

Our group has been investigating several aspects of this general problem, through the process of designing, developing, deploying and evolving three different web-based collaborative tools. Reflecting upon the process of developing these tools, and the lessons we learned through our experimentation with them, we have recognized the need for a systematic methodology for studying these systems. This methodology must (a) enable users to express interesting sociological questions, (b) provide computational support for systematic analysis of interesting community phenomena, and (c) enable the integrated analysis of information captured in different communities. These are the requirements driving the design of SociQL, which, to date, has been used to study four different communities.

This is clearly an active and fascinating area with a huge number of open questions and substantial opportunities for the development of innovative intelligent services. In the future, we plan to integrate SociQL to all three tools and to further expand its syntax and its analyses.

Acknowledgments. This work was generously supported by NSERC, iCORE, IBM, and was conducted in collaboration with B. Tansey, K. Bauer, M. Fokaefs, D. Chodos, D. Serrano and D. Barbosa.

References

1. Grudin, J.: Computer-Supported Cooperative Work: History and Focus. IEEE Computer 27(5), 19–26 (1994)
2. Grudin, J.: Groupware and Social Dynamics: Eight Challenges for Developers. Commun. ACM 37(1), 92–105 (1994)
3. Chodos, D., Naeimi, P., Stroulia, E.: A simulation-based training framework for health-science education on video and in a virtual world. Journal of Virtual Worlds Research 2(1) (April 2009)
4. Gutierrez, L., Stroulia, E., Nikolaidis, I.: fAARS: A Platform for Location-Aware Trans-reality Games. In: Herrlich, M., Malaka, R., Masuch, M. (eds.) ICEC 2012. LNCS, vol. 7522, pp. 185–192. Springer, Heidelberg (2012)
5. Stroulia, E.: Smart Services Across the Real and Virtual Worlds. In: Chignell, M., Cordy, J., Ng, J., Yesha, Y. (eds.) The Smart Internet. LNCS, vol. 6400, pp. 178–196. Springer, Heidelberg (2010)
6. Serrano, D., Stroulia, E., Barbosa, D., Guana, V.: SociQL: A Query Language for the SocialWeb. In: Kranakis, E. (ed.) Mathematics in Industry. Advances in Network Analysis and its Applications, vol. 18, pp. 381–406. Springer, Heidelberg (2013), http://dx.doi.org/10.1007/978-3-642-30904-5_17
7. Tansey, B., Stroulia, E.: Annoki: A MediaWiki-based Collaboration Platform. In: Web2SE: First Workshop on Web 2.0 for Software Engineering, ICSE 2010 (2010)
8. Espiritu, C., Stroulia, E., Tirapat, T.: ENWiC: Visualizing WIKI semantics as Topic Maps: An automated topic discovery and visualization tool. In: Proceedings of the 8th International Conference on Enterprise Information Systems, Paphos, Cyprus, May 23-27, pp. 35–42 (2006)
9. Hall, T., Strangman, N.: Graphic organizers, Wakefield, MA. National Center on Accessing the General Curriculum (2002), http://aim.cast.org/learn/historyarchive/backgroundpapers/graphic_organizers (retrieved October 2012)
10. Arazy, O., Stroulia, E., Ruecker, S., Arias, C., Fiorentino, C., Ganev, V., Yau, T.: Recognizing Contributions in Wikis: Authorship Categories, Algorithms, and Visualizations. Journal of the American Society for Information Science and Technology (JASIST) 61(6), 1166–1179 (2010)
11. Bauer, K., Fokaefs, M., Tansey, B., Stroulia, E.: WikiDev 2.0: Discovering Clusters of Related Team Artifacts. In: CASCON 2009, Toronto, Canada, November 2-6 (2009)
12. Fokaefs, M., Serrano, D., Tansey, B., Stroulia, E.: 2D and 3D Visualizations in WikiDev2.0. In: ERA track, 26th IEEE International Conference on Software Maintenance ICSM 2010, Timisoara, Romania, September 12-18 (2010)

13. Fokaefs, M., Tansey, B., Ganev, V., Bauer, K., Stroulia, E.: WikiDev 2.0: Facilitating Software Development Teams. In: 14th European Conference on Software Maintenance and Reengineering (CSMR 2010), Madrid, Spain, March 15-18 (2010)
14. Hasan, M., Stroulia, E., Barbosa, D., Alalfi, M.: Analyzing Natural-Language Artifacts of the Software Process. In: ERA track, 26th IEEE International Conference on Software Maintenance, ICSM 2010, Timisoara, Romania, September 12-18 (2010)
15. Boechler, P., Carbonaro, M., Stroulia, E., King, S., de Jong, E., Chodos, D.: Technical Challenges And Solutions For Creating Virtual Environments For A Health Science Interprofessional Course. The Internet Journal of Allied Health Sciences and Practice 9(4) (October 2011)
16. Chodos, D., Stroulia, E., Kuras, P., Carbonaro, M., King, S.: MERITS Training System - Using Virtual Worlds for Simulation-based Training. In: The International Conference on Computer Supported Education, CSEDU 2010, Valencia Spain, April 6-9,
17. Chodos, D., Stroulia, E., King, S.: Developing a Virtual-World Simulation. In: Third International Workshop on Software Engineering in Healthcare (SEHC 2011), ICSE 2011, pp. 71–78 (2011)
18. Chodos, D., Stroulia, E., King, S., Carbonaro, M.: A framework for monitoring instructional environments in a virtual world. British Journal of Educational Technology (accepted September 2012)
19. Chodos, D.: Using Virtual Worlds for Scenario-based Training. Ph.D. University of Alberta (Fall 2012)
20. Anagnostopoulos, A., Kumar, R., Mahdian, M.: Influence and correlation in social networks. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2008, New York, NY, USA, pp. 7–15 (2008)
21. Ganev, V., Guo, Z., Serrano, D., Barbosa, D., Stroulia, E.: Exploring and Visualizing Academic Social Networks. In: The 19th the ACM Conference on Information and Knowledge Management, CIKM 2010 Demo (2010)
22. Knorr-Cetina, K.: Sociality with objects: social relations in postsocial knowledge societies. Theory, Culture & Society 14(4), 1–30 (1997)