Enabling Service-Based Application Development through Social Objects

Peter D. Schott, Michael J. Burns, and R. Bruce Craig

Abstract—Software development is typically a social activity; development is most often performed by a team of people who must collaborate effectively. Application development enablers, such as web services and their APIs, can be viewed as social objects that can be interacted with just as one may interact with other people in the context of a social network. Extending enterprise social network applications and ecosystems to include API's and applications as first-class objects will have a significant impact on the social and technical processes through which enterprises develop service-based applications. In this paradigm application enablers can be tagged, followed, searched/discovered, and discussed across the enterprise development community. This paper presents a prototype framework architecture, design model, and usage scenario that exemplify this approach.

Index Terms—**application development, Enterprise 2.0, service oriented architecture, social network**

1 Introduction

Two forces are at work that will radically change the way software applications are developed in enterprises. The first of these forces is the rise of service oriented architecture (SOA) [1] or more specifically "Web Oriented Architecture" (WOA). [2, 3] With the broad proliferation of service oriented applications based on web services, software engineers are reaching outside their immediate teams to leverage services provided by others, both within their companies and outside (such as open source code and Google Code$^{(tm)}$), to enable effective development of their applications.

While the broad availability of web services present software engineers with many options to leverage in building the applications, when a developer needs code to perform a particular function, it is often easier to develop it herself than to go through the trouble of searching to find something suitable that might already exist. [4] Conversely, developers who want to expose services for use by others are often frustrated by the lack of an effective mechanism for broadly sharing their work.

The second trend that will change the nature of application software development is the emergence of enterprise social networking. At its core, enterprise software development is a social activity, requiring collaboration and interaction across organizational boundaries within an often global enterprise. [5, 6] It seems natural, therefore, to look for ways that social networking applications can assist with the social activity of software development. One such area is the effective exposure of application services Application Programming Interfaces (APIs) mentioned above.

Enterprise social networks and ecosystems can be extended to include services and applications as accessible objects. Doing so will have a significant impact on the social and technical processes through which enterprises develop service-based applications. This paper describes a model for viewing services, their APIs and the resulting applications as "first class" objects within a social network, thus enabling software engineers and other interested associates to interact with these objects as members of the social network. The key benefit is that services can be exposed and "discovered" in the social network more effectively than with current practice, which should dramatically increase reuse, shorten development cycles, and increase quality.

The remainder of this paper is organized as follows: Section 2 discusses the fundamental ideas for social networking in the context of interacting with services and applications. Section 3 presents a usage scenario to explore how services and applications can interact within the social network. Section 4 highlights conclusions and future directions for related work.

2 Enterprise Social Network Model

2.1 Define an Example Social Network Model

Our team has developed an enterprise social networking application, called People & Projects (P&P). We will use P&P as an example social networking application that can be extended to support the exposure of application services. P&P began as an experimental effort in Alcatel-Lucent Bell Labs to explore innovative enterprise social networking concepts. It has been in use across Alcatel-Lucent for approximately five months at the time of this writing.

Manuscript received December 8, 2009.

Peter D. Schott is with Alcatel-Lucent Bell Laboratories, Murray Hill, NJ 07974 USA (908-582-8163; fax: 908-582-8163; e-mail: pete.schott@ alcatel-lucent.com).

Michael J. Burns is with Alcatel-Lucent Bell Laboratories, Murray Hill, NJ 07974 USA (e-mail: mike.burns@ alcatel-lucent.com).

R. Bruce Craig is with Alcatel-Lucent Bell Laboratories, Columbus, OH 43213 USA (e-mail: bruce.craig@ alcatel-lucent.com).

The People and Projects social network data model exposes four primary social objects with which users can interact:

- People: Every Alcatel-Lucent employee and contractor is in the P&P database. Each person has his own Profile Page that displays information about that person. Some of the information is pulled from a corporate directory database; other information is added by users. Figure 1 shows an example People Profile Page from P&P.
- Projects: Projects represent the major work activities that people are performing. Any P&P user can add a new project to the database. Informal, as well as formal, projects can be accommodated in the P&P data model.
- Conversations: Conversations are public, text-based dialogues between P&P users. Any user can begin a conversation, and any other user can reply to the conversation.
- Tags: Tags are short descriptions of people, projects, or conversations in P&P. They provide a way for an individual user to describe an object to help her find the object later via P&P's discovery mechanisms. Because any user can tag any object, the set of tags for any person, project, or conversation represents the "collective wisdom" about that object. [7, 8]

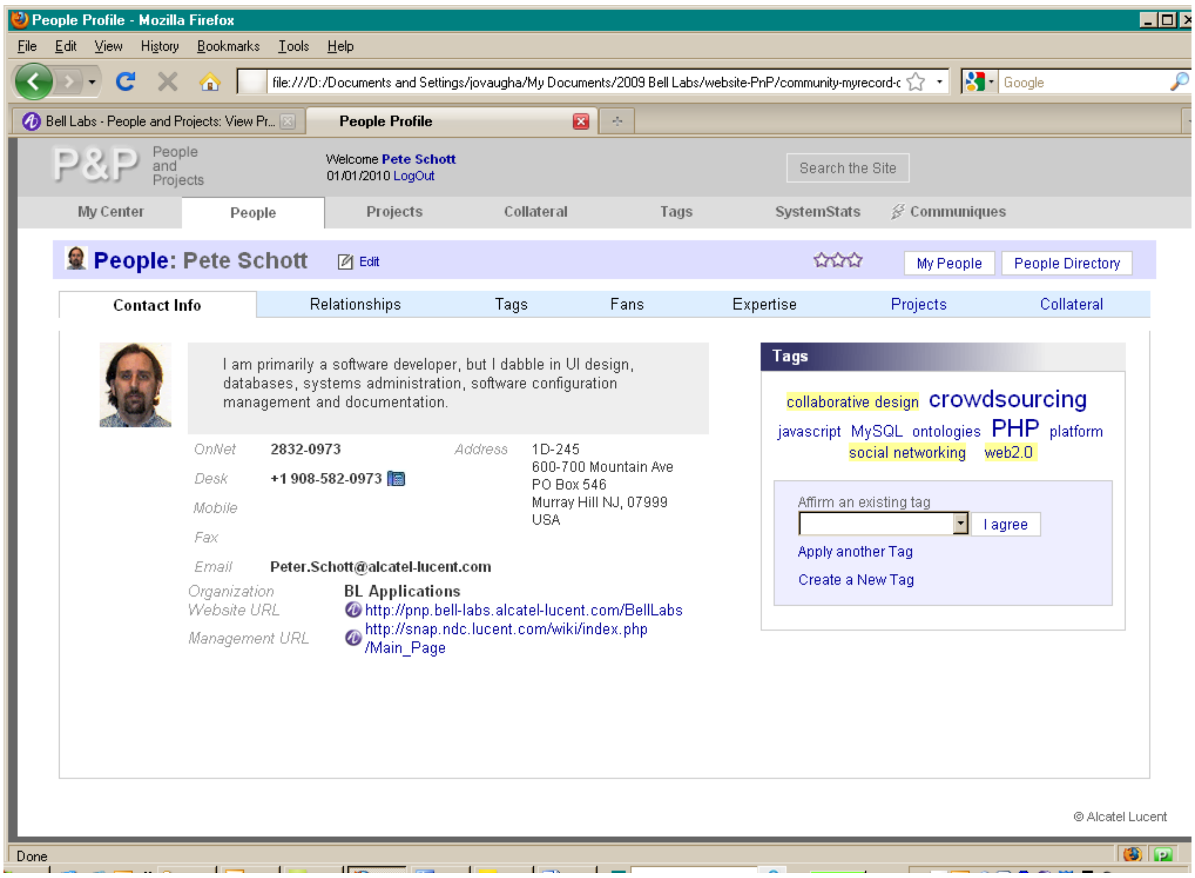
Users interact with these objects in several ways including:

- Tag: Apply tags to an object to describe the object in a meaningful context for the user.
- Follow: Denote that the user wants to monitor changes to the object, or, in the case of people objects, be informed when a person performs certain activities within P&P. When an object's properties change, the user is notified of these changes within P&P.
- Discuss: Have free-form conversations with other users. These conversations may reference other objects within P&P, such as projects.

These methods of interacting with the social objects are open to any member of the P&P social network as long as they are logged into P&P.

Figure 1 shows an example Profile Page for a P&P user. It includes a short description about the user's skills and interests, contact information about the user, and tags that have been applied for this person. Other tabs on this pages show more detailed information about this person.

When a P&P user interacts with another P&P object in any of these three ways, P&P is able to infer social relationships through those interactions. For example, P&P users may be related because one user has tagged another user. Similarly, the act of following a person or project or conversation causes a "following" relationship, and conversing about an object causes a "discussing" relationship.



I am primarily a software developer, but I dabble in UI design, databases, systems administration, software configuration management and documentation.

OnNet	2832-0973	Address	1D-245 600-700 Mountain Ave PO Box 546 Murray Hill NJ, 07999 USA
Desk	+1 908-582-0973		
Mobile			
Fax			
Email	Peter.Schott@alcatel-lucent.com		
Organization	BL Applications		
Website URL	http://pnp.bell-labs.alcatel-lucent.com/BellLabs		
Management URL	http://snap.ndc.lucent.com/wiki/index.php/Main_Page		

collaborative design crowdsourcing javascript MySQL ontologies PHP platform social networking web2.0

Figure 1: Example People Profile Page

Second-order relationships are also exposed in P&P. For example, two users might share the same tag or be following the same object. Cross-object second-order relationships are also possible. For example, a person and a project might be related because they share the tag "Java" because they are both working in the Java language. These second-order relationships are a key part of P&P. The exposure of second-order relationships is how P&P allows users to "discover" other users who they share interests with or projects on which they might want to collaborate.

2.2 Service APIs and Applications as Social Objects

The above social networking model can be extended to include software development objects, such as service APIs and applications, as first-class objects in a social network. Viewing service APIs and applications as social objects is a significantly different approach from existing service-based development approaches used today. Such an approach offers important benefits.

The first benefit is that services and service-based applications will be searchable and discoverable within the social network. Just as users can search for other people or projects in a social networking application like People & Projects, they can search for services and applications, based on "traditional" data about these objects such as title and description. However, including these objects in the social networking framework opens up the possibility of using a much richer set of data to help discover services and applications. These additional data or attributes include tags applied to the services and applications by their developers and by other social network community members who have knowledge of the services and applications, perhaps through their experience in using them. The additional data also include metadata about which applications are consuming which services, which developers have contributed to which services and applications, how recently services and applications have been created and modified, etc.

In addition, exposing services and applications through a social network enables members of the community to "follow" services and applications that may be of interest to them, just as they can follow people and projects that are of interest. In this way, they can be notified immediately when there are changes or updates to the software development objects that are important to them.

Another benefit of this approach is the ability for the social networking application to foster discussion about the software development objects of interest. For example, if one developer has a question about a particular service, he can easily contact the service's developer through the social network. He could also opt to post a question about the service to the social networking community, in which case anyone who is following the service in question would see the question and be able to respond.

Second-order relationships, as described above for people and projects, also are valuable when thinking about services and applications in this context. Tags shared by services, applications, and developers enable shared second-order relationships between those objects to be discovered. Similarly, they enhance the potential for conversations between a wider set of the community. For example, if one developer is following the tag "call control" and a service has been tagged with "call control," that developer will be notified about changes to that service or conversations about that service, even though she is not following that service directly. Thus, these second-order relationships are key to exposing services and applications much more broadly.

Current approaches in building service-based applications require the developer to either build the applications based on services from a single source or the developer must use his own expertise and social acumen to find and understand appropriate services to utilize. Many times developers rely on searching developer-focused forums and news lists, email, and other informal channels to find pertinent information. Once the application is built, unless the developer has some support arrangement with the service provider, typically at a cost, there are few effective methods for the developer to be informed of changes to the service.

Viewing services and applications as first class social objects has the potential to enable a fundamental paradigm shift in service-based application development. This paradigm shift will be realized through the ability of software engineers and other social network members to discover, interact with, and leverage services and applications within the context of a social network. Further, through interacting with members of the social network, the availability and functionality of the services and applications will be exposed, discovered, and evolved based on the social interaction of the user community. A major part of the paradigm shift that we envision comes about because exposing services and applications as social objects provides a social perspective and provides important new social tools for software developers to use in the inherently social work that they do. These tools give them a way to unify the collaboration that is necessary in software development, but which is currently handled through disjointed mechanisms.

2.3 Services and Applications Data Model

This section describes a model and approach for extending a social networking application to include software development objects of interest.

Services and applications have many attributes that can be exposed to interested parties. This paper does not attempt to detail all the potential attributes, but instead highlights and discusses attributes that are central to viewing services and applications as objects within a social networking ecosystem.

A service object contains data and metadata that allow the service to be searched and discovered by others and that provide sufficient information about the service so that the service is usable by application developers.

The service data model contains:

- Service name: The name of the service.
- Service description: A short description of the primary operation(s) performed by the service.
- Service owner(s): Name(s) of the developer(s) who created the service and/or who are currently maintaining the service. These names would include links to the developers' profile pages in the social networking application.
- Service documentation: Typically a link to any available in-depth documentation about the service. This documentation may be stored in the social networking application itself or may be in a corporate data store external to the social networking application.
- Service definition: Definition of the service with example usage and service parameters (if any).
- Service location: URL for the service location.
- Service creation date: Date the service was registered with the social networking application.
- Service version history: Audit trail of changes made to the service since it was registered with the social networking application.
- Tags: Tags that have been applied to the service by its developer(s) and others in the social networking community who have knowledge of the service
- Applications using the service: Applications that consume the service, including a link to each application's profile page in the social networking application (if the application has been registered with the social networking application).
- Developers using service: Names of developers in the social networking community who are using the service, including links to their profile pages in the social networking application.
- Service discussion: Threaded conversations about the service among members of the social networking community.
- Service Followers. Social networking community members who have expressed an interest in being notified about changes made to the service object.

The data model for applications is essentially the same as the above model for services, but with the application as the central object, rather than the service. In addition, one data element for the application would provide services used by the application (and links to the profile pages for those services), instead of the "Applications using the service" data element mentioned above for services. Additionally, since applications can be deployed or installed on one or more systems or "nodes," therefore the application model will contain an attribute that lists the nodes the application is deployed to.

Services (and applications) are visualized in the social network application through dynamic service (and application) profile pages that present the data and metadata listed above for a given service (or application). The profile page is dynamic in the sense that as new services are added into the social networking application, their profile pages will be created dynamically, with no additional coding effort required. Also as attributes for the services and applications change, the changes will be reflected on the profile pages.

Figure 2 shows an example Service Profile Page. This page shows basic information about the service including title, description, owner, and URL to access the service. Tags that have been applied to the service are also shown. Other tabs provide more detailed information about the service.

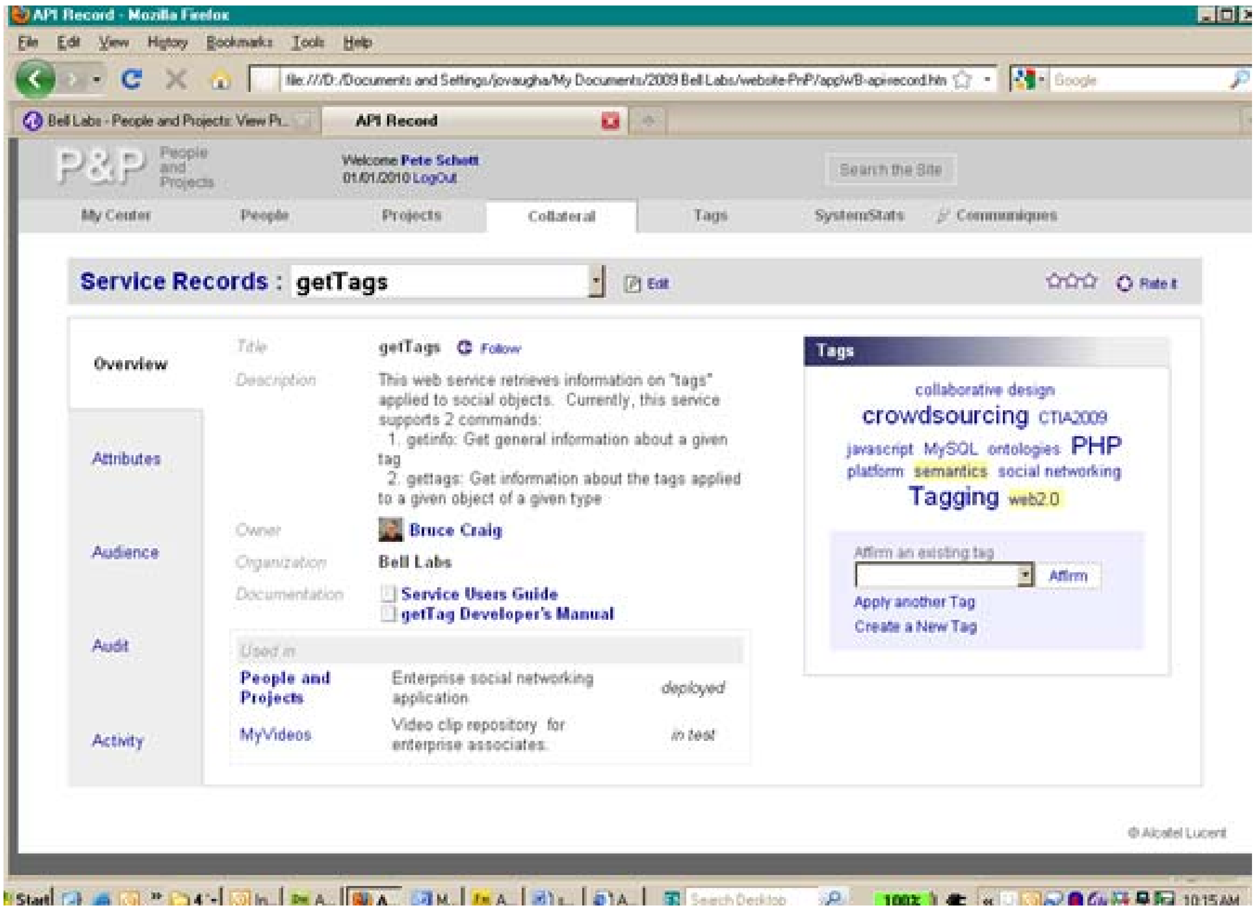
Enabling the visualization of services and applications within a social network, while important, does not by itself enable members of the community to easily discover and utilize these services. For that to happen, the services must be fully participating "members" of the community. This is realized through allowing users to discover services through the social network search facility and, most importantly, to interact with services through the social networking relationship objects – tagging, following, and discussing. Since tags and discussions themselves are discoverable as objects in the social network, they allow members to discover services through second-order relationships as well.

The ability to follow service objects is especially important because it enables the social networking community to engage in the life cycle of the service and participate in the service's enhancements and quality. Through following a service, a community member can monitor changes in the service, tags applied to the service, applications and developers using the service, and discussions about the service. Followers are "pushed" the updates and are then aware of changes as they occur.

Figure 3 shows the new communiqués that have come in to alert this user that a service he is following is being followed by another user as well as a new tag that has been applied to a feedback conversation he is following.

2.4 Developer's Service Cart

The "Developer's Service Cart" helps developers in working with services. It is a container object in which a developer can store references to services in the social networking application. It serves essentially as a way of "checking out" services that the developer is considering using in an application they are developing for easy reference while they are being incorporated into the application the developer is coding.



file:///D:/Documents and Settings/jovaugha/My Documents/2009 Bell Labs/website-PnP/appWB-apirecord.htm

Welcome Pete Schott
01/01/2010 LogOut

Service Records : getTags

Title	getTags
Description	This web service retrieves information on "tags" applied to social objects. Currently, this service supports 2 commands: 1. getinfo: Get general information about a given tag 2. gettags: Get information about the tags applied to a given object of a given type
Owner	Bruce Craig
Organization	Bell Labs
Documentation	Service Users Guide getTag Developer's Manual

Used in		
People and Projects	Enterprise social networking application	deployed
MyVideos	Video clip repository for enterprise associates.	in test

Tags

collaborative design crowdsourcing CTIA2009 javascript MySQL ontologies PHP platform semantics social networking Tagging web2.0

Affirm an existing tag
Apply another Tag
Create a New Tag

Figure 2: Service Profile Page

A typical scenario for using the "Developer's Service Cart" is:

1. A developer visits a service's profile page that they want to use,
2. The developer checks the "Add to developer's cart" check box option. By doing this, the service's information will be copied into the cart's container object for viewing later by the developer within their development environment.
3. As the developer builds their application, they can view the service's information from the cart.

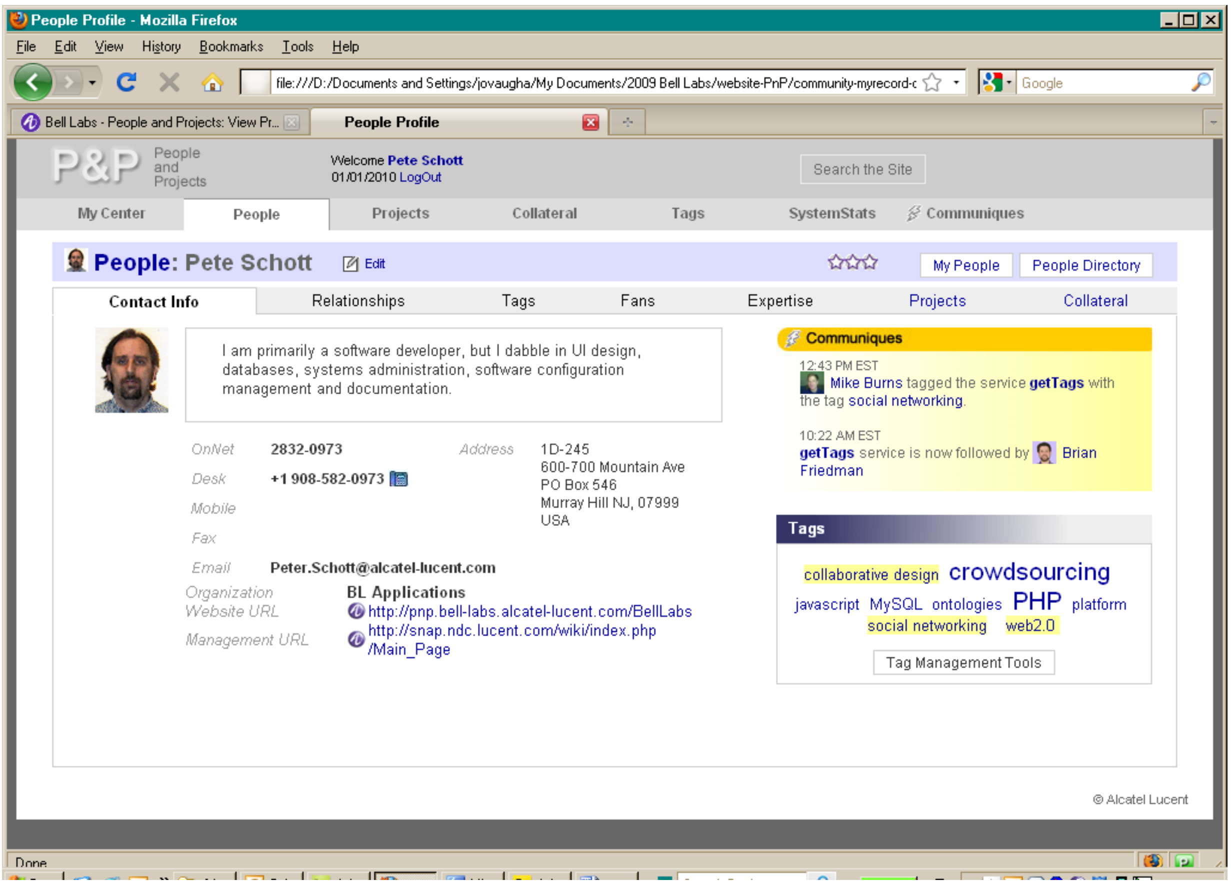
A large number of services can be added to the Developer's Service Cart, and they persist in the cart until the developer explicitly removes them. Each developer can have her own instance of the cart, which is visible only to them. A reference to a service in the cart contains a link to that service's profile page in the social networking application.

3 Usage Scenario

This section brings the services and applications enabled through social objects process to life through a usage scenario that highlights a potential use for this approach. The user's goal in this scenario is to develop a new service-based application, called ConfSocial, that will enable employees in her company to view conference papers along with detailed information about the authors and to apply tags to the papers. This example scenario explores how such a service-based application can be enabled by leveraging services visualized as objects in a social network.

This scenario builds on the People and Projects social networking application described earlier. In addition to the social objects described in Section 2, People and Projects provides web services that allow access to the people, projects, tags, and conversations object data and metadata. These services are exposed as social objects within People and Projects.

This scenario assumes that the enterprise has a central repository for conference papers (ConfDB) that contains basic information about the papers and their authors. The data in ConfDB is exposed through web services that are accessible by application developers within the enterprise. These services are not modeled as social objects but are defined on the enterprise's development wiki pages.


People Profile - Mozilla Firefox

File Edit View History Bookmarks Tools Help

file:///D:/Documents and Settings/jovaugha/My Documents/2009 Bell Labs/website-PnP/community-myrecord-c

Bell Labs - People and Projects: View Pr... | People Profile

P&P People and Projects

Welcome Pete Schott
01/01/2010 LogOut

Search the Site

My Center | People | Projects | Collateral | Tags | SystemStats | Communiques

People: Pete Schott Edit

My People | People Directory

Contact Info | Relationships | Tags | Fans | Expertise | Projects | Collateral

I am primarily a software developer, but I dabble in UI design, databases, systems administration, software configuration management and documentation.

OnNet	2832-0973	Address	1D-245 600-700 Mountain Ave PO Box 546 Murray Hill NJ, 07999 USA
Desk	+1 908-582-0973		
Mobile			
Fax			
Email	Peter.Schott@alcatel-lucent.com		
Organization Website URL	BL Applications http://pnp.bell-labs.alcatel-lucent.com/BellLabs		
Management URL	http://snap.ndc.lucent.com/wiki/index.php/Main_Page		

Communiques

12:43 PM EST
Mike Burns tagged the service getTags with the tag social networking.

10:22 AM EST
getTags service is now followed by Brian Friedman

Tags

collaborative design crowdsourcing javascript MySQL ontologies PHP platform social networking web2.0

Tag Management Tools

© Alcatel Lucent

Done

Figure 3: Communiqué Bar showing new notifications

3.1 Usage Scenario: Current Application-ConfApp

A service-based web application exists, called ConfApp, that allows users to view conference papers stored in ConfDB by author. ConfApp stands alone and is not integrated with other applications or social networks. ConfApp uses services provided by the enterprise to query the conference paper repository by enterprise associates and reply with papers that have been authored by associates matching the query.

ConfApp users are also members of the enterprise's "People and Projects" social networking application, and these users realize there is much more information about the authors of conference papers that a new application could present to those viewing the papers.

As starting step in building the new application, ConfSocial, a software engineer is assigned to determine how to add more author information to the application. In addition, users have requested the ability to "tag" papers of interest enabling them to identify the papers in a more personal manner and find them more easily later.

3.2 Usage Scenario: Investigate Enhancement Potential – Discover Available Services.

The software engineer is also a member of the People and Projects social network, and knows that People and Projects treats services and applications as viewable objects in the social network. So the software engineer visits the People and Projects' services application "store." The application "store" is a search facility the allows users to search for applications "registered" with it by using keywords to match on tags, application data, conversation data, and people data. After doing a search, the software engineer finds there are no matches. This indicates that no application that meets the developer's needs has already been developed and registered with People & Projects. The software engineer expected this result, but started her work with this search anyway just to be sure such an application did not already exist. The software engineer also understands that there is a likelihood there are services available to provide the data required, even though the needed application itself does not exist yet.

The software engineer knows People and Projects supports a similar "store" for services that have been developed by the various business units and the research departments. The business units and research organizations have already added many services to Services store so that they are modeled as objects within the People and Projects social network as described in section 2.2.

The software engineer visits the service "store" and does a search using keywords that could return services that would be useful for the application the software engineer wants to develop. For example, she might search for “authors ‘get people info‘ ‘create tags’”. In her search criteria she includes terms to find technologies she wants to use, in this case “PHP”. The search returns a listing of several services that meet the search criteria based on service data such as service titles, services descriptions, tags, and conversation text.

In this case, among the query results, People & Projects services for “getting people information,” “finding tags,” and “applying tags” are returned. Future work, will focus on introducing a recommendation approach to presenting the results. A recommender “engine” will view services metadata and developer profile information and suggest services based on analysis of the search results. The software engineer then visits each service’s profile page to explore the details of the services.

3.3 Usage Scenario: Services Object Profile Page

The software engineer visits the service object profile pages of the three services to get information about the services to determine whether they are applicable for the new application.

In the case of the “getting people information” and “finding tags” services, the information presented on the service profile page makes it clear that the services do match the requirements and the software engineer adds the services into her Developer’s Service Cart. However, the “applying tags” service description raises some questions so the software engineer decides to start a conversation on the service profile page, hoping she can get the questions answered. When a visitor engages in conversation about a service, all owners of the service and all community members who are following the service, the service owner, or tags that have been applied to the service will be notified of the discussion and can participate in the conversation with the visitor. Additionally, all conversations are logged and are reviewable by community members. In short order, the questions are answered and the software engineer adds the “applying tags” service into the “Developer’s Service Cart.”

At this point, the software engineer has the required services to build the ConfSocial application to include more author information and enable tags to be applied to the conference papers. The software engineer develops the required code to access the services, leveraging the information contained in the “Developer’s Service Cart” in her development environment.

The new conference paper application now includes the additional author information and allows for tagging of the papers. It is determined this application should be made available across the enterprise. To achieve this effectively, the application needs cross-organization exposure. Typical tools for advertising applications include broadcast emails and postings to blogs or wikis. A drawback of these methods is that once the initial announcement is made, associates who may have missed it won’t know about it. Also, email search is awkward at best. In People & Projects, an application can be made easily discoverable by registering the application with the People and Projects application store. This action allows applications to be searched and discovered similarly to services discovery described in section 3.2. Knowing this, the software engineer decides to register the application with the People and Projects applications store.

3.4 Usage Scenario: Register Application

To register the enhanced conference paper application, the software engineer visits People and Projects Applications Store. Here when the software engineer is presented with the Application store home page, she selects “Register Application” and her “developer” profile page is presented displaying personalized information for that developer including:

- Applications that developer has registered, with links to each application’s profile page
- Services that developer has recently viewed
- Services that developer is following
- That developer’s Developer’s Service Cart

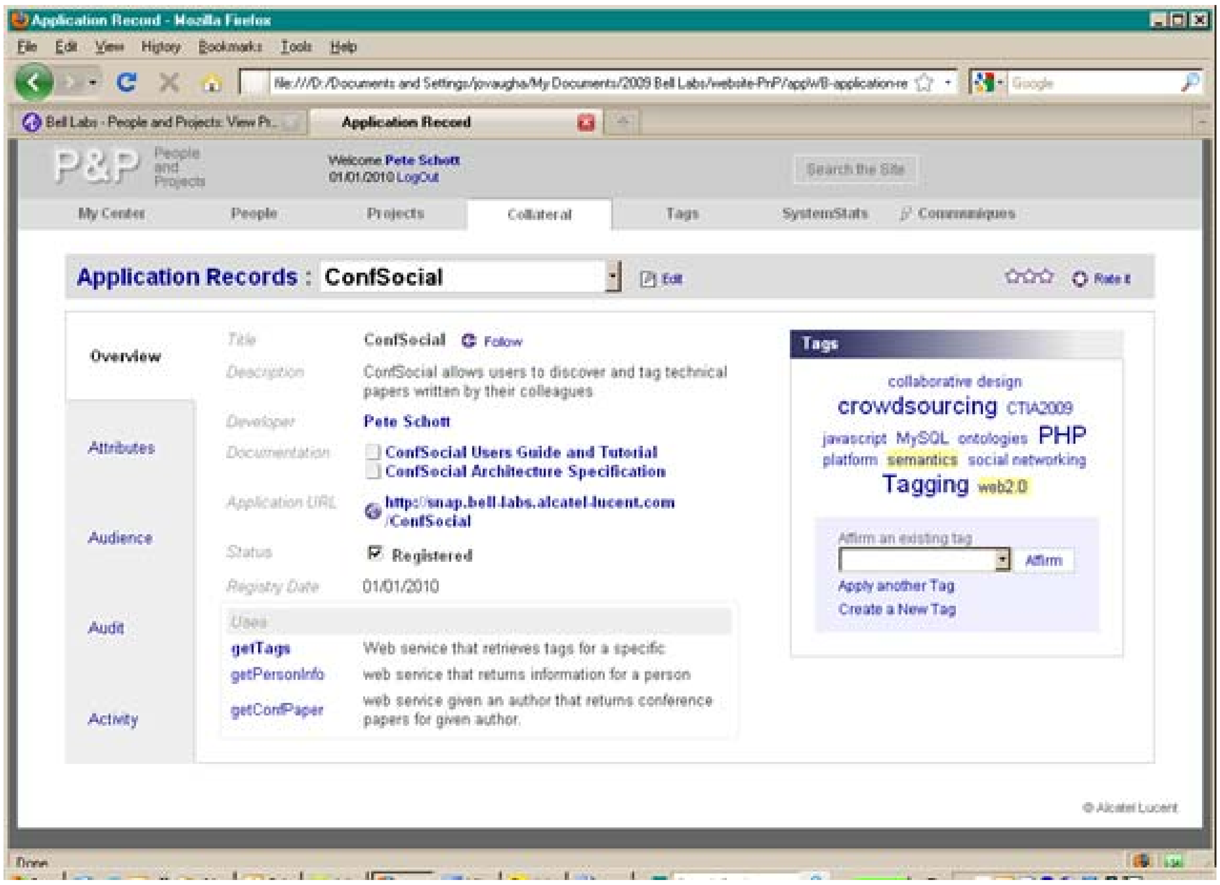
In addition the page includes a button that the developer can click to register a new application

Since the software engineer wants to register the conference paper application, the software engineer presses the “Register Application” button and a wizard guides her through the registration process.

The registration process includes collecting information to instantiate the new application social object in the social network, including:

- Application name and description.
- Application location (URL).
- Services used by the application (can be added from the Developer’s Service Cart).
- Tags to apply to the application

After the developer completes the wizard, the application is registered and can be discovered using the various methods supported by People and Projects.



Application Records : ConfSocial

Title	ConfSocial
Description	ConfSocial allows users to discover and tag technical papers written by their colleagues
Developer	Pete Schott
Documentation	ConfSocial Users Guide and Tutorial ConfSocial Architecture Specification
Application URL	http://snap.bell-labs.alcatel-lucent.com/ConfSocial
Status	☑ Registered
Registry Date	01/01/2010

Uses	
getTags	Web service that retrieves tags for a specific
getPersonInfo	web service that returns information for a person
getConfPaper	web service given an author that returns conference papers for given author.

Figure 4: Application Profile Page

Figure 4 shows Application Profile Page for the ConfSocial application after the application has been registered. It includes title, description, owner, and other key information about the application. Tags that have been applied to the application are also shown. Users can follow this application by clicking on the “follow” link.

If the application is later tagged or followed by other community members, the application’s software engineer will be notified. Additionally, the software engineer can be notified of application usage. If anyone in the community is currently following the software engineer, they will be notified that she has registered the application.

3.5 Usage Scenario: Discovering the Application

After the application is registered in People & Projects, the social network community can discover the application in several ways that leverage the application being exposed as a social object. These include searching for the application by tags, keywords, or developer. Community members will also be notified about second-order relationship events that are relevant to the application, so they might discover the new application because they are following one of the tags that has been applied to the application. Like other social objects in People & Projects, the application can be followed by social network community members. Followers of applications will be notified when the application is tagged, attributes are changed or conversations about the application are started or replies made to previous conversation threads

3.6 Usage Scenario: Conversing with/about Applications

Each application and service object profile page has an area that provides visitors with the ability to start or join conversation threads pertaining to the application. The conversations are themselves a first class social object. Therefore community members who are following the application or tags that have been applied to the application or people who participate in the conversation will be notified about the new conversation. The conversation can be viewed by community members and they can participate in the conversation. This enables the community to share and explore discussions on application topics across the enterprise, increasing the potential for effective collaboration around similar topics.

4 FUTURE WORK

Future work will focus on providing a mechanism for effective discovery of existing services that are outside social networks so they can automatically be registered as new objects in the social network. Other work will focus on observing patterns of service use, for example what services are used together and in what sequence so that the platform can recommend services that should be considered based on services the user already has expressed interest in. This recommendation engine could use fuzzy linguistic modeling as described in [9].

ACKNOWLEDGMENT

The authors wish to acknowledge the important contributions to People & Projects and to many of the concepts described in this paper made by Brian Friedman and John Vaughan.

REFERENCES

[1] D. Garlan, "Software Architecture: a Roadmap," International Conference on Software Engineering, Proceedings of the Conference on The Future of Software Engineering, 2000, pp. 91 – 101.

[2] D. Linthicum, "Web-Oriented Architecture (WOA) Gains Momentum, *Web 2.0 Journal*, May, 2008. Available: http://web2.sys-con.com/node/560954.

[3] D. Hinchcliffe, "What Is WOA? It's The Future of Service-Oriented Architecture (SOA)," Dion Hinchcliffe's Blog - Musings and Ruminations on Building Great Systems, February, 2008 Available: http://hinchcliffe.org/archive/2008/02/27/16617.aspx.

[4] K. Sherif and A. Vinze. "Barriers to adoption of software reuse: A qualitative study, " *Information & Management*, Volume 41, Issue 2, December 2003, pp 159-175.

[5] R.Spencer, "The streamlined cognitive walkthrough method, working around social constraints encountered in a software development company," Conference on Human Factors in Computing Systems, Proceedings of the SIGCHI Conference on Human factors in Computing Systems, 2000 pp. 353-359.

[6] A. McAfee. "Enterprise 2.0: The Dawn of Emergent Collaboration," *MIT Sloan Management Review*, Volume 47, Number 3, 2006, pp.21-28.

[7] S. Farrell, T. Lau, E. Wilcox, S. Nusser, and M. Muller, "Socially Augmenting Employee Profiles with People-Tagging," Symposium on User Interface Software and Technology Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST), 2007, pp. .91-100.

[8] C. Li and J. Bernoff, *Groundswell: Winning in a World Transformed by Social Technologies*, Boston, MA: Harvard Business School Press, 2008.

[9] C. Porcel, A.G. Lopez-Herrera, E. Herrera-Viedma, "A Recommender System for Research Resources based on Fuzzy Linguitic Modeling," *Expert Systems with Applications: An International Journal*, Volume 36, Issue 3, 2009, pp. 5173-5183.