Towards an OpenID-based solution to the Social Network Interoperability problem

Position paper for the W3C Workshop on the Future of Social Networking

Michele Mostarda
`michele@asemantics.com`

Davide Palmisano
`davide@asemantics.com`

Federico Zani
`federico@asemantics.com`

Simone Tripodi
`stripodi@asemantics.com`

Abstract

The main aim of this paper is to present the OpenPlatform, the Asemantics OpenID solution developed for several Italian web portals. Starting from the developers' experience in the field of digital identity, this work also describes the Global Social Platform as an open-standard based solution to the Social Networks interoperability problem. Basically, it consists of a platform allowing users to aggregate their social graphs that are spread over a bunch of different social networks. The OpenID main concepts, such as the user Identity Page, are the foundational ideas behind the Global Social Platform since they can be considered as the unique place where information is stored. A set of Java API is provided in order to write Adapters and Converters that are, mainly, Global Social Platform plugins allowing third party developers to pull data from different data silos and showing them in several different formats.

1 Introduction

According to the classical widely used definition in the fields of sociology and anthropology, the term 'Social Networks' was generally used to refer to complex sets of relationships between members of a specific social system. Graph-theoretical approaches[1], in which such social systems were described using well-known mathematical structures called graphs, were widely used as theoretical frameworks for the sound proof of a lot of algorithms. These methodologies show all their expressive power especially in the sociometric clique problem resolution where, intuitively, the need to find a highly cohesive subgroup of individuals is reduced to the problem of finding a subgraph[2] against a set of metrics defined on the whole graph.

In this sense we can map the clique theory to the concept of the social graph as defined in [3] where the author informally defines it as *"the global mapping of everybody and how they're related"*. This statement, jointly with the vision of a "web of data" driven by Semantic Web technologies, allows us to make several positive assumptions about the feasibility of Giant Global Graph[4] aware applications development.

In fact, the recent vibrant proliferation of social network web applications is putting the social graph concept at the centre of the scene: users are projecting parts of their identities in several different containers each of them with different data models for the social graph representation.

Typically a user's own social graph might be subdivided into several overlapping subgraphs stored and handled into different containers, also known as "data silo" which keep this information in a closed manner, or by exposing tools for accessing them through RESTful APIs. Approaches to the solution of what is known as *"The Walled Gardens problem"*, can be divided into two main classes:

API standardization: comprises all efforts such as the Google OpenSocial APIs[5]or Yahoo! Social APIs[6] that are aimed at solving the problem by providing the specification of a common API. The main idea is that the social applications which adhere to the API specification can be accessed in a uniform fashion, having complete portability through different social networks.

Mash-up applications development : ad-hoc applications that use several different APIs from different social networks in order to enhance the functionalities of an existent one. Applications like updating Facebook's status through Twitter, for example, can be considered part of this class of solutions. They address only a narrow set of specific requirements using different approaches for different problems.

The main idea behind our solution is to consider:

"OpenID as the hammer that allows us to break down these walls",

due to the functionalities that can be built on its main concepts.

More specifically,

- the user's **Identity Pages** can be used as main repositories of user information. For example, their relationships can be represented using FOAF profiles and stored within their Identity Page, while their general information can be encoded using Microformats following the Data Portability philosophy.

- The **Single-Sign-On** approach provided by **OpenID** can be applied to several social network applications so improving user experience in an *open fashion*. The power of OpenID can be assessed from the point of view of two different applications that are supporting that technology (as Plaxo and identi.ca) if we see a user login operation . The logged user has different social graphs (in terms of his relationships) in the two different containers, but the usage of a unique OpenID account allows him to be recognized as a single individual in order to merge (or, more appropriately mash-up) his social graphs.

This work is mainly intended as an introduction to the *Asemantics Open Platform* that is, basically, an extensible and easily deployable implementation of the OpenID Specification, and an hypothesis of its extension called *Global Social Platform* aimed aimed at solving the *Walled Gardens Problem* using our implementation of the OpenID Specification. The paper is organized as follow: In section 2 we will describe the *Open Platform*, an OpenID based solution developed by this Company, while in section 3 we will give a brief overview of the *GlobalSocialPlatform*, a research project based on the Open Platform, going towards the Global Giant Graph.

2. The Asemantics OpenPlatform

Company experience in developing *OpenID* solutions can be summarized in a general architecture named *Open Platform* which consists in the implementation of both a *Provider* and a *Relying Party* service; in addition, OpenPlatform provides several auxiliary components specifically aimed at increasing system security and reliability.

2.1 Architecture

A load balancer (usually hardware) acts as a system front-end, distributing requests on a battery of J2EE Web Containers (typically Apache Tomcat 5.x). Every Web container is accessed through an Apache Server 2.x used to perform URL rewriting, resource access restrictions and HTTPS support.

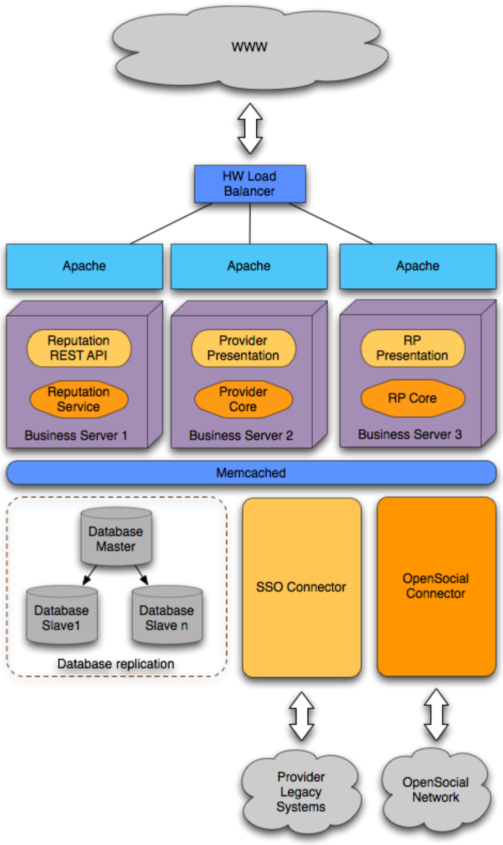

Figure 1: Open Platform Architecture

Web containers host both presentation and REST services. The presentation layer is based on JSTL solutions. Specific J2EE frameworks are not used, with the exception of an IoC (Inversion of Control) support based on Google Guice. The DAOs (Database Access Object) are written with iBatis.

The Business Tier lies on persistence and connector layers. Persistence layers are divided into a storage persistent infrastructure and a memory persistent infrastructure.

The storage persistent infrastructure is based on the MySQL master-slave replication and is intended to maintain persistent data like User profiles, preference logs and related information.

The memory persistent infrastucture is a *Memcached* service used to share user sessions across Web Containers.

The connector layers provide access to the legacy Single Sign On systems (SSO Connector) and to Social Networks (OpenSocial Connector). Access to Social Networks is done through the Open Social and Social Graph API. The social data retrieved with the OpenSocial Connector is used to generate contents in user Identity Pages. Examples of such contents are FOAF profiles and user widgets. With the Open Social API the Identity Page can work as a Social Container.

Access to the existing SSO service and user database is done through the SSO Connector module.

Figure 1 depicts the whole Open Platform architecture.

2.2 OpenID Provider

The Provider implements the authentication on RPs through the *Single-Sign-On* general approach. In respect to the OpenID philosophy, the *Identity Page* is central to user characterization . The User can define several profiles, in each of which he can expose different sets of attributes with different visibility. These attributes are visible in the User Identity Page, every information set is expressed with a compatible Microformat to enable automatic recognition by browsers and crawlers.

User privacy is granted during data exchange by providing control of required attributes. The User is notified of the list of attributes that RP requires about him, and he can choose whether to complete or abort the data transition. The User can specify default grants for RPs that can be modified at any time with the specific preferences panel. User security is pursued with several functionalities. The platform supports the logic for black and white listing of RPs.

2.3 OpenID Relying Party

The Relying Party development of *Open Platform* is straightforward. From the user point of view there is a *SREG* and *AX* fully compliant implementation, and the administrator has access in monitoring and control with the well-known approach of black and white listing.

2.4 Reputation system

The Platform is able to provide reputation feedback to their own users concerning Relying Parties or Identity Providers using pluggable *Reputation Services* like KarmaSphere [8].

It is possible to activate policies forbidding authentications with RPs having a reputation score under a given threshold. We are evaluating the convenience of giving a RP reputation score based on *Search Engine Ranking*.

2.5 Anti phishing

A phishing attempt might be represented as a classic man-in-the-middle attack: the idea is to impersonate a well-known website, so as to fool the user that he's safe to write down important information (username, password and other sensible data).

The more the OpenID adoption grows, the more the OpenID itself becomes a bigger fish to catch, since a single account may be used for hundreds of websites. This alone makes OpenID more vulnerable, since losing one password means you've lost them all.

Open Platform adopts several antiphishing technical solutions, from SSL certificates, to browser plugins and more, but the main focus has been kept on letting the platform consumer decide which approach is most suitable for his environment.

3 Global Social Platform: an Open-Platform extension

In this section, the *Global Social Platform*, an extension of the *Open Platform*, is presented. It consists mainly of a platform that acts as a proxy layer between a user with his own OpenID account, and several social networks with which he wants to interact, in order to solve the so-called *interoperability problem*. The main idea here is to provide a platform that exposes two kinds of RESTful APIs: the first ones is aimed at wrapping several different APIs provided by the existent social networks (either OpenSocial-compliant or not), while the second one can be used by services consumers. Basically, the platform provides some features that allows the mashing-up of data from different sources (i.e. social networks the user belongs to).

3.1 Motivations and first concepts

Several blogs and articles have already introduced the well-known *Social Networks Interoperability Problem* and showed how OpenID and social networks are quickly converging , like the excellent one, cited below, provided by Dare Obasanjo[7], which can be enough to motivate the problem:

I have a Facebook profile while my fiancè has a MySpace profile. Since I'm now an active user of Facebook, I'd like her to be able to be part of my activities on the site such as being able to view my photos, read my wall posts and leave wall posts of her own. I could ask her to create a Facebook account, but I already asked her to create a profile on Windows Live Spaces so we could be friends on that service and quite frankly I don't think she'll find it reasonable if I keep asking her to jump from social network to social network because I happen to try out a lot of these services as part of my day job. So how can this problem be solved in the general case?

Before introducing how the design of our *Global Social Platform* tries to address this problem, some informal, but necessary definitions are provided.

With regard to Figure 2, we define:

Social Network as a web application that allows the management of user profiles and, at least, exposes a set of APIs for accessing the data. Apart from those social networks that adhere to the *OpenSocial Specification*, we consider as part of this definition also such web applications that, usually, are not considered as social networks. For example, even if the main aim of microblogging applications is different from that of Facebook, they also expose a set of APIs permitting access to the users data and their relationships.

Global Social Aggregator is the core of the Global Social Platform. Informally it is an engine that is able to run pieces of software called *Adapter* and *Converter* written using a specific set of Java APIs. Basically, it is a container that allows users to:

- login with an OpenID account,
- define which Social Networks they belong to providing the access information,
- filter the data, telling the system which information from which different Social Networks will be visible or not,
- show such information on his Identity Page, customizing it using widgets.

This middle layer is able to access the content of social sites using generic adapters, mashing up the collected data and returning them in a manner that enables their encoding in Identity Pages. These set of functionalities are achieved through our platform's main components:

Data Model is the internal data model. All the data pulled from every different social network is mapped in this internal representation and, then, stored on a relational database. At the moment we are studying the feasibility of using RDF for the internal data representation.

Global Social Platform Java API is a set of Java classes that allows platform extension. It gives the developers tools for writing new Adapters and new Converters.

Adapter Is a piece of software, written using the Global Social Platform Java API, that has the responsibility to map all the data obtainable from one specific Social Network API to the internal Data Model. Basically an Adapter acts as a wrapper: it allows the pulling of data from an existent container and making them available to other Platform components.

Content Filter The access to the Data Model is filtered by this component. User-defined policies are translated in rules that will be executed by Filters letting them define restrictions to the visibility of their resources. Based on OAuth[9], this component allows the user to limit access to pieces of his social graph in a way that is publicly accessible and is machine readable.

Converter Is the module that defines rules and actions to be done in order to map data represented in the internal data model in one available formats like FOAF, XFN, JSON or even exposed with an Open Social API built on top of it.

Social Browser Is any tool, be it a service or a desktop application, able to support at least one of the formats provided by converters in the middle tier.

Social Consumer Represents anybody having a digital identity (simply, an OpenID account) and using the Global Social Platform to access his own different profiles spread over all the various Social Networks he belongs to.

Component and data flows are shown in Figures 2 and 3 that summarize the concepts described above.

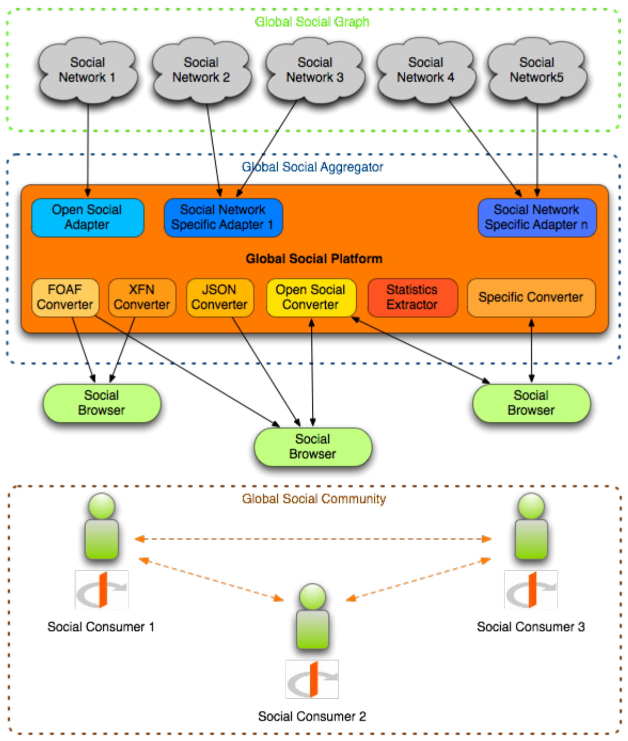

Figure 2: Global Social Platform Architecture

3.2 Main Requirements

The main idea behind the design of this platform derives from considering the possibility of keeping the coupling between several different APIs available today and the platform itself at a low level. Even if the *OpenSocial Specification* represents a positive effort towards the interoperability of social networks, the problem of having an extensible and low coupled architecture still remains. Our adapter-based solution is aimed at preserving the overall platform integrity from external environment changes: as the external APIs change, it will be sufficient to update the relative Adapter, avoiding rewriting the whole architecture.

Apart from the fulfillment of classical software systems requirements, such as modularity and extensibility, the main Global Social Platform aim is to provide a framework able to meet the requirements clearly declared by Brad Fitzpatrick in his famous post[3]. Next section underlines how the Global Social Platform addresses these issues.

3.3 The key concept: Adapters and Converters

The Platform provides a set of Java APIs meant for the development of pluggable *Adapters* and *Converters*. Identity Page URL acts as a unique identifier for the users. In this way, each node in the Data Model that represents a User account on various Social Networks can be encoded as links to a node representing the User's Identity Page. From the point of view of the developer *this enables an abstraction that solves the Node Equivalence problem*. Once each user is identified by their Identity Page URL, the API provides a basic mechanism for the retrieval of all the incoming and outgoing edges to that node representing, for example, friendship relations. For each relationship the Data Model can encode its source: the Social Network where the links really exist. Basically, a developers can write Converters with the high level of abstraction offered by the Data Model. On the other hand, portable contacts can be easily achieved just by simply writing new Converters.

In Figure 4 a brief overview of this architecture is shown.

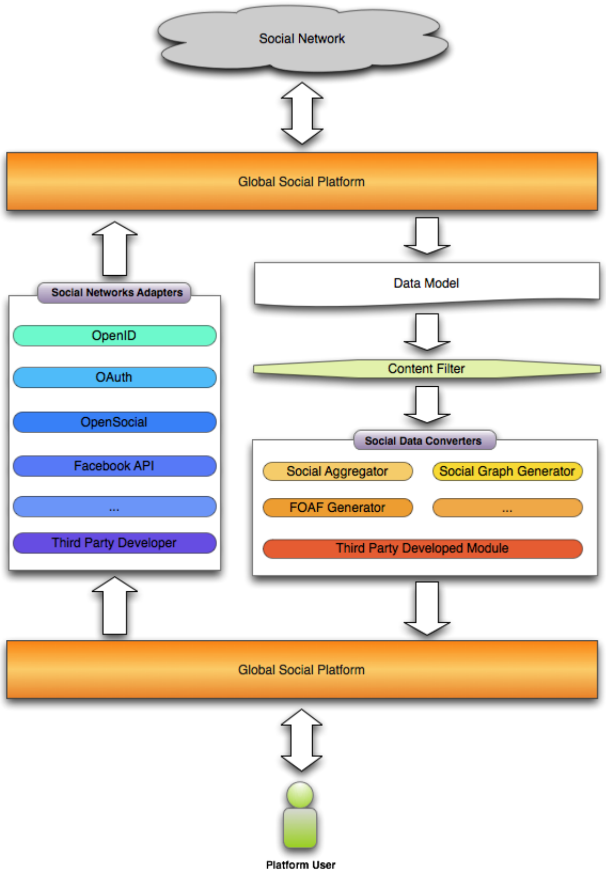

Figure 3: Global Social Platform data flows

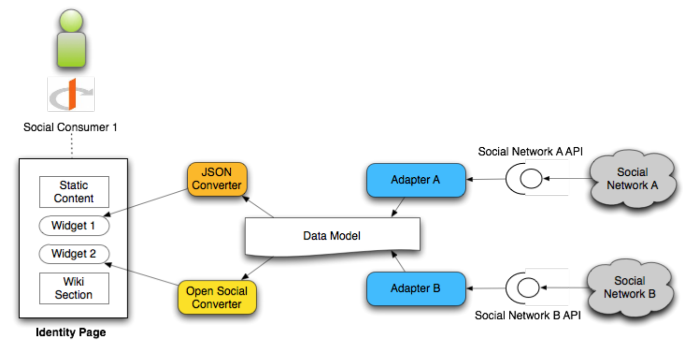

Figure 4: Global Social Platform: Adapters and Converters

4. Conclusion

4.1 Results

The OpenPlatform has been developed in collaboration with one of the major players of the Italian telecommunication context. The Platform is still under testing, but several modules are already publicly accessible. We are planning together with our partners the development program for the year 2009, where the Global Social Platform will become a reality.

4.2 Future developments

We're following the evolution of several promising social frameworks, in particular *Shindig*[10], an Apache Open Source project whose target is to provide a reference implementation of the *Open Social Specification*. Our purpose is to integrate a social framework with a portlet container, as pointed out by *SocialSite*[11], but with a more compliant approach.

Other topics to be explored in the next months will be connected with the integration of RDF inside the platform. The first intention is to define an extension of the OpenID Attribute Exchange with the support of RDFa[12] and provide a reference implementation.

References

[1] Barnes, J.A. 1983. "Graph theory in network analysis", *Social Networks*, 5: 235-244.

[2] Alba, Richard D. 1973. "A graph-theoretic definition of a sociometric clique" *Journal of Mathematical Sociology*, 3: 113-126

[3] Brad Fitzpatrick, "Thoughts on the Social Graph", http://www.bradfitz.com/social-graph-problem/

[4] Tim Berners-Lee, "Giant Global Graph", http://dig.csail.mit.edu/breadcrumbs/node/215

[5] http://code.google.com/apis/opensocial/docs/0.8/spec.html

[6] http://developer.yahoo.com/social/

[7] Dare Obasanjo, "A Proposal for Social Network Interoperability via OpenID", http://www.25hoursaday.com/weblog/2007/08/13/AProposalForSocialNetworkInteroperabilityViaOpenID.aspx

[8] http://karmasphere.com

[9] http://oauth.net/

[10] http://incubator.apache.org/shindig/

[11] https://socialsite.dev.java.net/

[12] http://www.w3.org/TR/xhtml-rdfa-primer/