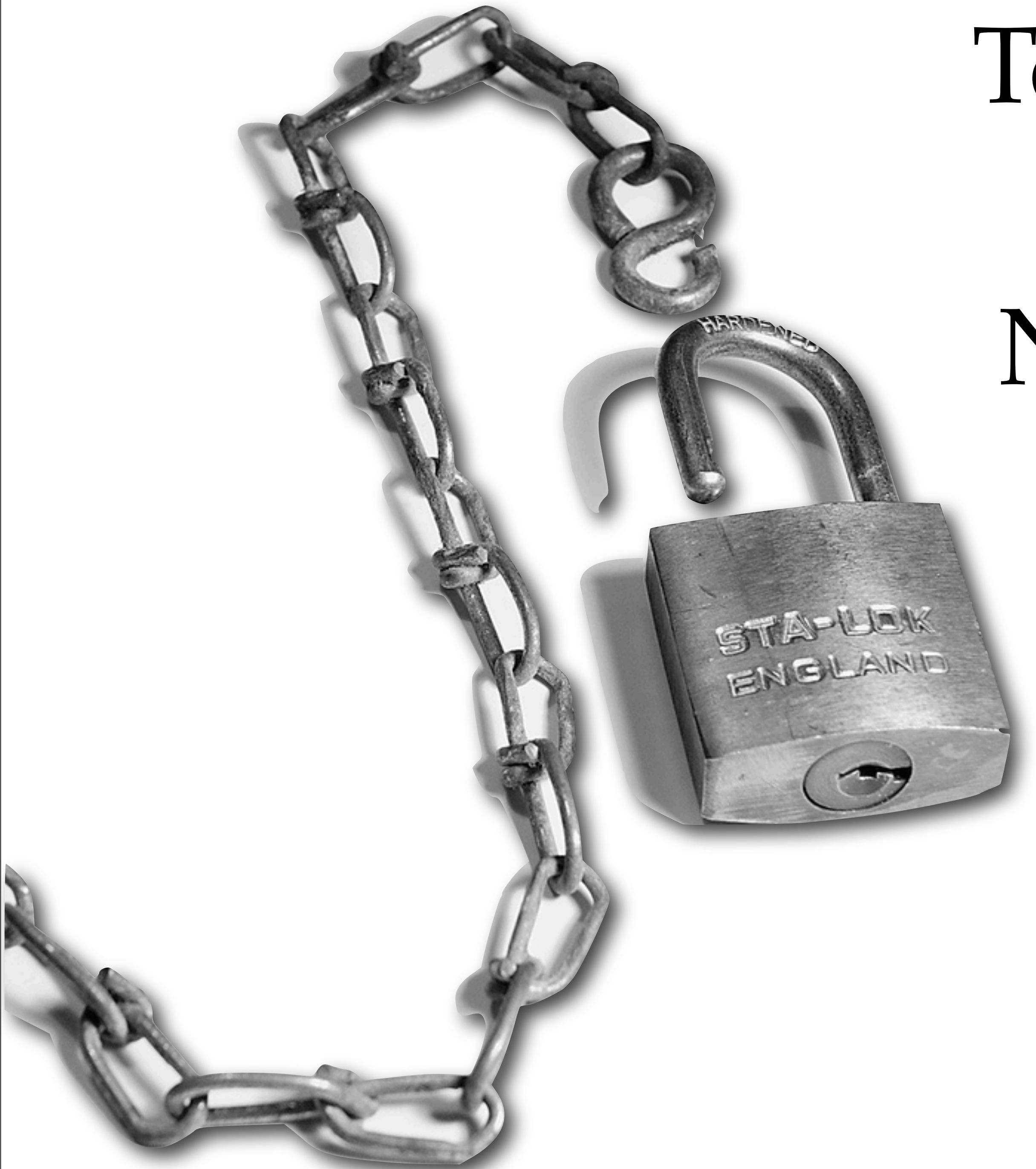
W3C Workshop on the Future of Social Networking, Barcelona January 2009



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

M. Mostarda, D. Palmisano, F. Zani, S. Tripodi

About Us

- We are proud member of the W3C and of the *OpenID* Europe and USA Foundations
  - focusing on Semantic Web activities
- We have developed both *OpenID* provider and relying party for several Italian Telecommunication and Media Corporates
  - Past and present costumers include Skype, Joost, BBC

Contents

- about us
- problem statement
- the *Global Social Platform* solution
  - architecture
  - adapters and converters: the Java APIs
  - data model
- a simple use case
- conclusions and future developments

Problem statement

- *"Walled gardens"*, Social network interoperability problem, data silos, social network portability ...
  - different terms for the same class of problems
- How to create, manage and use the information contained in the users own social graph
  - in a silo-independent manner,
  - with a mechanism for the user unique identification and
  - with the possibility of a fine-grained privacy rules definition

Problem statement

- As the email systems in the early '90 we are now dealing with a *"balkanized"* domain[1]
- Is our opinion that market trends in recent analysis are still encouraging social networks operators to keep their "walled gardens"
  - in this sense the *OpenSocial* initiative can be seen only as standard way to "expose" the data in a uniform fashion
  - pull-push through REST APIs is **the** application development paradigm

Evan Prodromou, *"OpenMicroBlogging"* this Workshop

Problem statement

- On the other hand, we believe in the role of *OpenID* as

  "the hammer that allows us to break down these walls"

- If we consider:
  - a user's Identity Page URL as his unique identifier
  - the Identity Page itself as the main entry point where a user can access and show to the world his aggregated data

The GSP solution

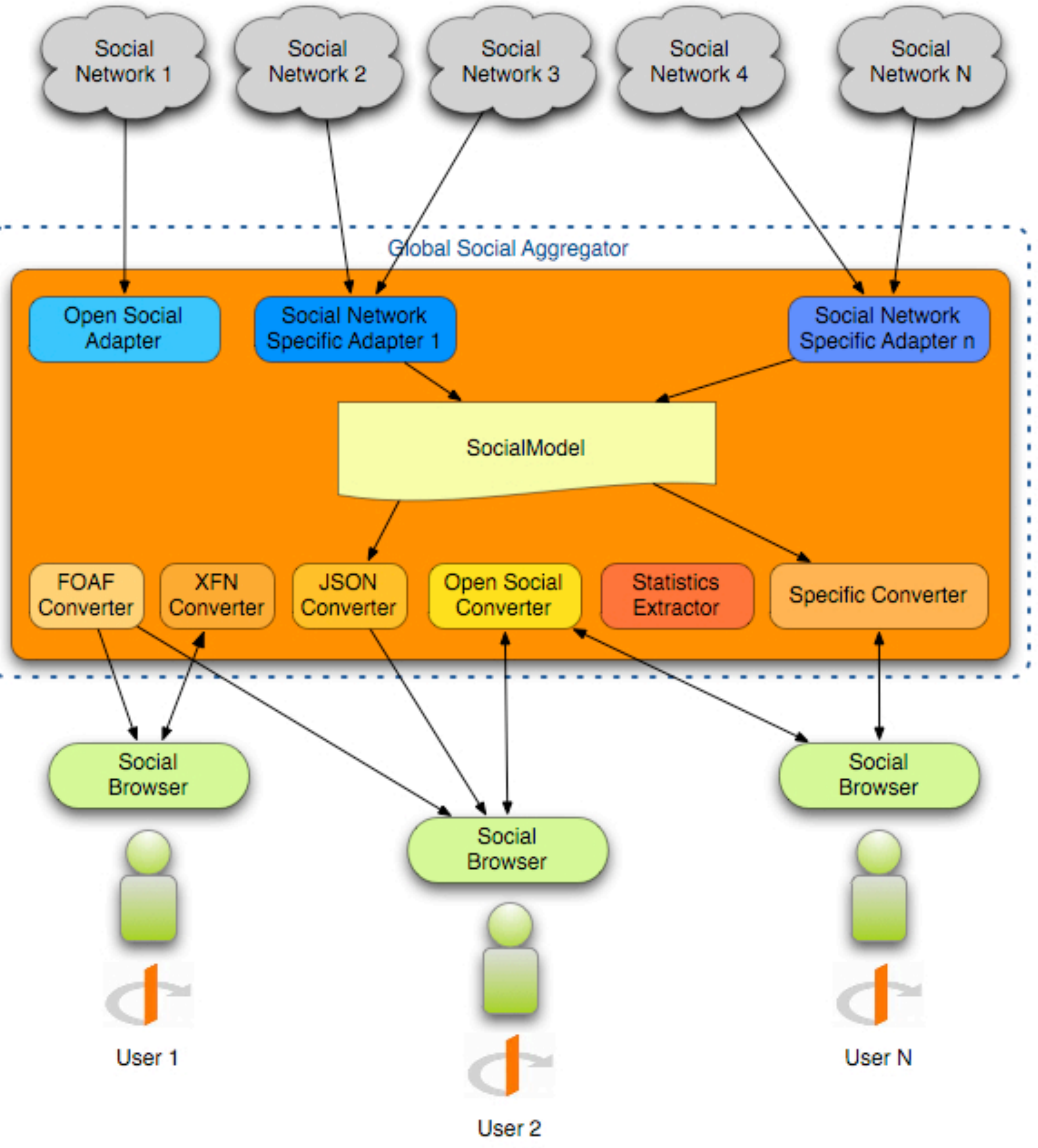
- Starting from this consideration we are currently designing a Java environment for the deployment of social applications that:
  - will access the users social graphs independently from the specific silo it is partitioned
  - aggregate, manage, and export such data with other formats
  - can refer to the users using an unique identifier
- informally, a social network **aggregator** Java engine

The GSP solution

- Mainly, the *Global Social Platform* is an architecutre that provides:
  - a engine that can runs pieces of software (*Socialets*):
    - *Adapters, Converters*
  - a set of Java APIs for writing *Socialets*
  - a data model able to represent the users social graph and their activities in a uniform fashion
    - developers can deal with user data without regarding on which silo they are contained

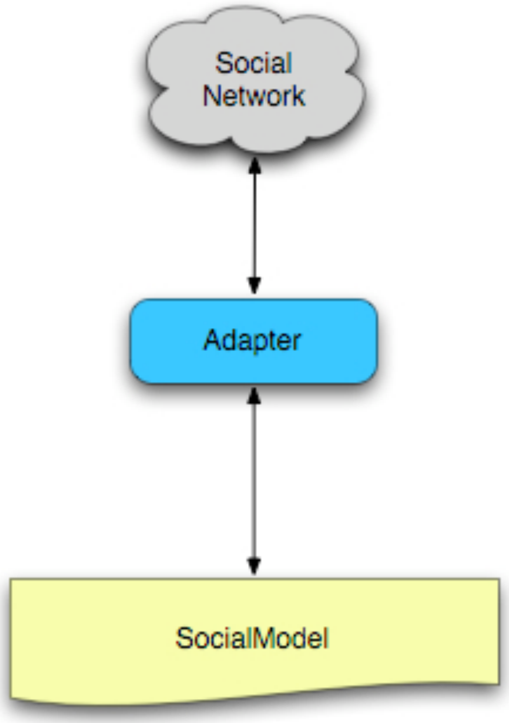
The GSP solution

- architecture outlook
  - Socialets
    - Adapters
    - Converters
  - Social Model
    - an hybrid persistence component



The GSP solution

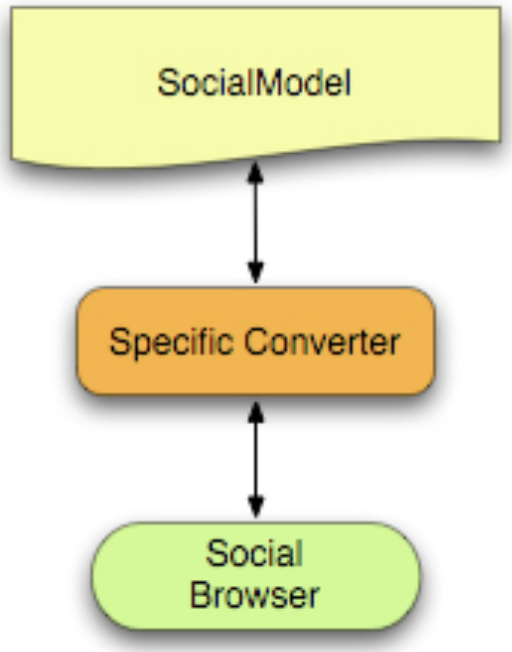
- Adapter
  - *Socialets* that acts as bridges between one specific Social Network API and the data model
  - basically, a REST API wrapper



```
/**
 * Contains the bridging logic between the {@link com.asemantics.socialaggregator.SocialModel} and
 * a <i>Social Network</i>.
 */
public abstract class Adapter extends Socialet {

    public abstract void marshall(SocialModel socialModel, AdapterRequest adapterRequest, AdapterResponse adapterResponse);

    public abstract void unmarshall(SocialModelChange socialModelChange, HTTPClient client);

    public void process(Request request, Response response, SocialModel socialModel) {
        ...
    }
}
```

The GSP solution

- Converter
  - This *Socialets* can read and write from the data model and has the responsibility to expose such data in several different way



```
/**
 * Contains the bridging logic between the {@link com.asemantics.socialaggregator.SocialModel} and
 * the exposed API.
 */
public abstract class Converter extends Socialet {

    public abstract void convert(ConverterRequest converterRequest, ConverterResponse converterResponse,
         SocialModel socialModel);

    public void process(Request request, Response response, SocialModel socialModel) {
         ...
    }

}
```

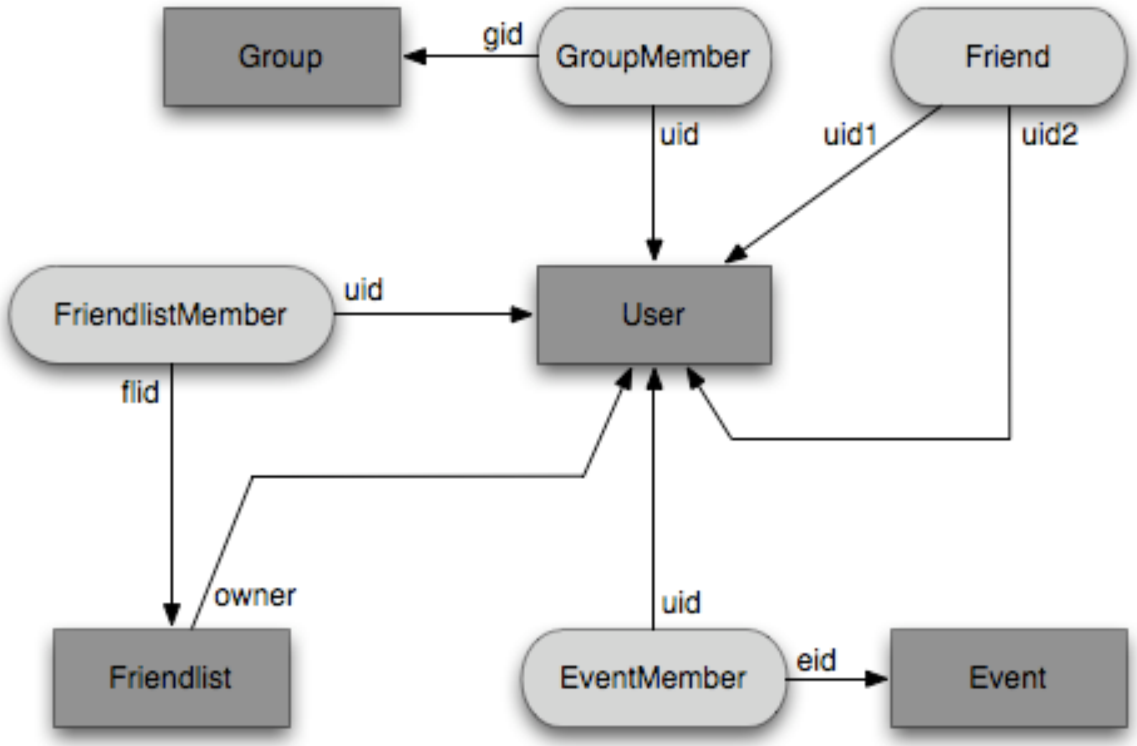
The GSP solution

- Social Model: the internal repository
  - a relational schema modeled as the union between the *OpenSocial* data model and the Facebook one
  - the main intent is to obtain a data model with an information capacity[1] that is the union of the two
    - one of the thousand possible compromises between flexibility and reliability
    - everything that can be accessed through OpenSocial or FB can fit in it

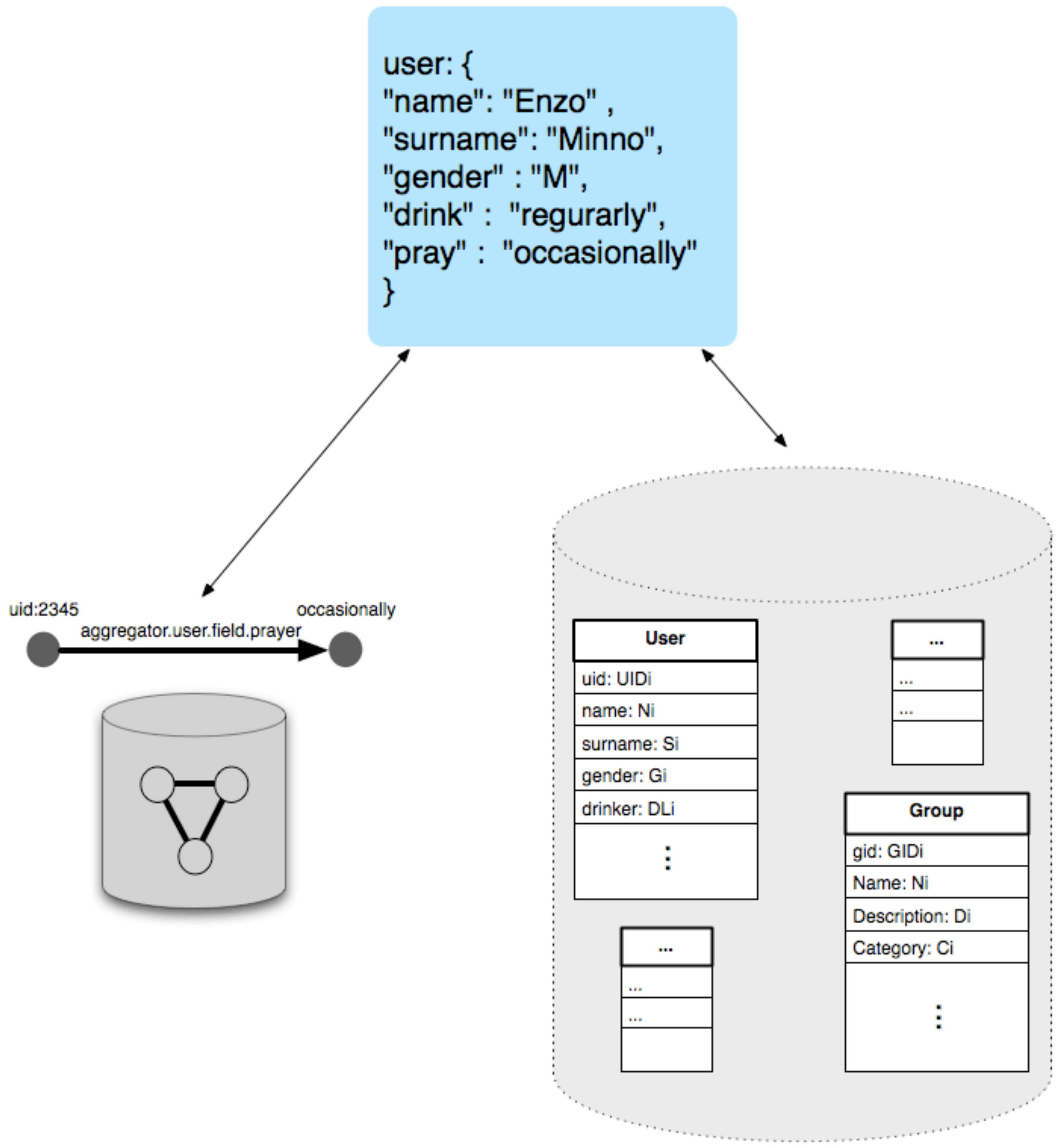
Richard Hull, *"Relative information capacity of simple relational database schemata"*, Proceedings of the 3rd ACM SIGACT-SIGMOD symposium on Principles of database systems

The GSP solution

- Social Model: the internal repository
  - as stated before, the Social Model has an hybrid nature that allow its extension using RDF
  - Entities, relationships and attributes that doesn't fit in the relational model will be expressed as triple and stored separately



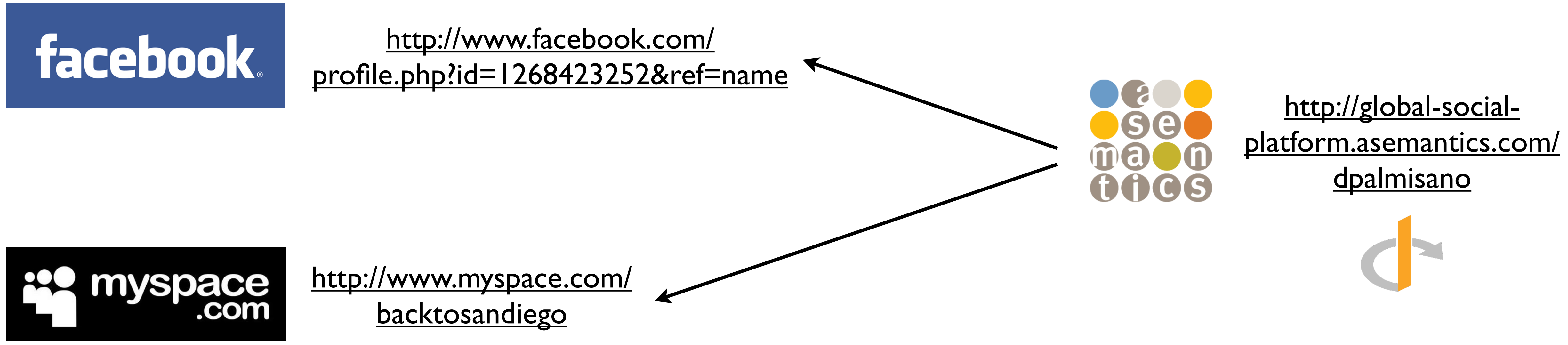
The GSP solution



```
user: {
"name": "Enzo" ,
"surname": "Minno",
"gender" : "M",
"drink" :  "regurarly",
"pray" :  "occasionally"
}
```

A simple use case

- Let’s image a user with two account
  - one on Facebook and one on MySpace


facebook
http://www.facebook.com/profile.php?id=1268423252&ref=name
myspace.com
http://www.myspace.com/backtosandiego
asemantics
http://global-social-platform.asemantics.com/dpalmisano

  - and that such user want to be able to update both status simultaneously from a Java Script Widget and to show it on his Identity Page

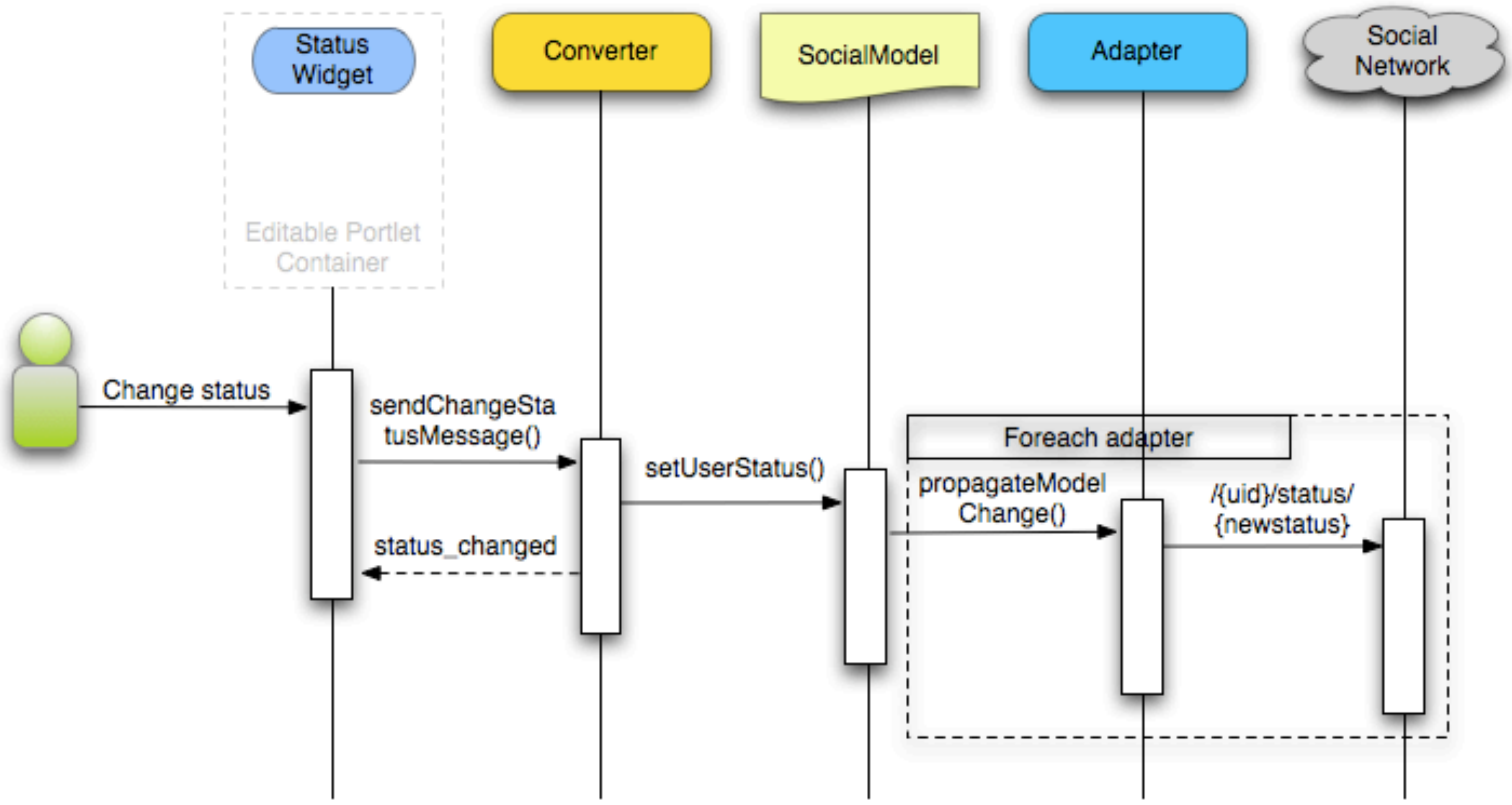
A simple use case

- From the developer's point of view:
  - In the **best** case:
    - a Facebook and MySpace Adapters have been already written and
    - a suitable Converter (that expose a REST interface) has been implemented;
    - then it would be sufficient to add them to the GSP engine, write a simple JavaScript Widget (that calls the Converter) and then make it available for end users

A simple use case

- From the developer’s point of view:
  - In the **worst** case:
    - Facebook and MySpace Adapters have to be written,
    - a suitable Converter (that expose a REST interface) need to be implemented, add them to the GSP and write the final JavaScript Widget
  - The excess of coding can be shared for further applications since Adapters and Converters can be used by different applications
    - changes in a top-level Social Network API can be embraced just updating its own Adapter

A simple use case



Conclusions

- We are close to the end of the inception phase
  - terminology and requirements need to be consolidated,
  - some architectural components are in a prototypal level,
  - looking for partners
- What's the difference with the recent Facebook Connect?
  - just one,

Conclusions

- We are close to the end of the inception phase
  - terminology is not consolidate,
  - some architectural components are in a prototypal level,
  - looking for partners
- What's the difference with the recent Facebook Connect?
  - just one,

our solution is intended to work across different silos....

Thank you!