OpenSocial
A standard programming model for the Social Web

Facebook Developer Meetup
February 13 2008, Mountain View

Patrick Chanezon
Developer Advocate
chanezon@google.com

What does Social mean?

Eliette what do you do with your friends?

WE
LOOK
AT
EACH
OTHER.

TALKING

LAUGHING


WE
HELP
EACH
OTHER!

WE
READ
TOGETHER!

WE
DO PROJECTS
TOGETHER.

Jaiku’s Jyri Engeström's 5 rules for social networks: social objects

1. What is your object?
2. What are your verbs?
3. How can people share the objects?
4. What is the gift in the invitation?
5. Are you charging the publishers or the spectators?

http://tinyurl.com/yus8gw

How do we socialize objects online
without having to create yet another social network?

Making the web better by making it social


plaxo


friendster


facebook


LinkedIn


hi5


viadeo


Ning


myspace.com
a place for friends


orkut
beta


LinkedIn

API

Friends
Profiles
Activities


orkut
LinkedIn
hi5
salesforce.com


orkut
LinkedIn
hi5
salesforce.com

OpenSocial

A common open set of APIs for building social applications across multiple sites

This is NOT GoogleSocial.
It’s about making the Web more social, not just Google.


orkut
LinkedIn
hi5
salesforce.com
OpenSocial


orkut
LinkedIn
hi5
salesforce.com
OpenSocial


Ning
plaxo
friendster
XING
hi5
orkut
beta
viadeo
six apart
mixi
ミクシィ
mixi, Inc.
salesforce.com
Success On Demand.
LinkedIn
ORACLE
天際網
www.tianji.com
ENGAGE
bebo
imeem
Hyves.net
always in touch with your friends
OpenSocial
THEIKOS
Delivering Results On-Demand
RockYou!
FotoFlexer
virtual Tourist
Real Travelers - Real Info
iLike
Flixster
slide

Standards-based

html+javascript

How does it work?

10 minutes to an OpenSocial app

One API, Many Websites

One API

- client-side JavaScript - version 0.7 ready for production
  - standard Web development tools: HTML + Javascript
  - server optional
- server-side REST (not specified yet)
  - Google proposal based on Atom Publishing Protocol
  - standard XML schema

Many Websites

- every OpenSocial website exposes the same API

==> more users for every app

==> more apps for every user

Core Services

- People ("who I am")
- Friends ("who I know")
- Activities ("what I'm doing")

- Persistence (state without a server)

OpenSocial API changes

0.5 -> 0.6 -> 0.7

Review of 0.6

0.6: the environment

- `hasCapability`: is a function available in the current container?
- `supportsField`: is a field available on a user profile?
- `getDomain`: where am I running?

0.6: navigation

- `requestNavigateTo:` from one page to another
- `getSurface`: profile, canvas, etc.?
- `getSupportedSurfaces`: what's available in this container?

0.6: improved permissions

- restrictions on when viewer data can be accessed, e.g. owner gadget does not always get to know about the viewer
- `requestPermission`: can I please see viewer data?

0.6: secure content fetching

- `makeRequest`: signed data requests
- POST as well as GET

0.6: activities

- no more `Stream`
- no more folders
- no more summary (title and body are sufficient)

0.6: profiles

- can check for `supportsField`
- profile data requested with explicit field list instead of broad categories (BASIC, FULL)

Intro to 0.7

(from the priority list)

0.7: core gadget specification

- as previously discussed, "core" gadget functionality is being opened under the same terms as OpenSocial
- 0.7 will clarify which (non-OpenSocial) APIs are part of the specification
- everything needed for core gadget compliance will be in Shindig
- functions re-namespaced as `gadgets.*`
  - e.g. `IG_Adjust_Frame_Height`, etc.

0.7: viral growth

- `requestShareApp`: with user's authorization, send messages to the user's friends about this app
- `requestSendMessage`: with user's authorization, send an app-specific message to another user

0.7: standardized person fields

- location
- organizational affiliations, schools, jobs
- picture, profile song, profile video
- status, "about me", interests
- gender, sexual orientation, relationship status
- date of birth, ethnicity
- children, pets
- music, movies, tv, books, activities, sports
- turn ons, turn offs, romance, looking for

0.7: environment

- methods to determine locale, language, country, etc.
- methods to determine surface geometry
- initial methods or CSS styles to allow gadgets to match container look & feel

0.7: activity templates

- HTML with simple variable substitution syntax
- Defined as messages in the gadget spec
- Developers set custom key/value pair parameters when posting an activity. Parameters can be referenced for variable substitution
- Activity Summaries will consolidate multiple events into a single item in the activity stream.
- Example message bundle:

```
<messagebundle>
  <msg name="LISTEN_TO_THIS_SONG:Artist">{$Viewer.Count} of your friends
have suggested listening to songs by {$Artist}!</msg>
  <msg name="LISTEN_TO_THIS_SONG:Song">{$Viewer.Count} of your friends
have suggested listening to {$Song}!</msg>
  <msg name="LISTEN_TO_THIS_SONG:Viewer">{$Viewer.DisplayName} has
recommended {$Song.Count} songs to you on ListenToThisSong.</msg>
</messagebundle>
```

0.7: simplification of data APIs

- no global app data
  - feeds are a more general solution
  - can be prefetched for performance
- no instance app data
  - can be implemented on top of person data
  - open issue: hints to container about "joinability"?

0.7: specification of multiple surfaces

- gadget XML extended to support multiple `<Content>` blocks
- each block declares the surfaces it should appear on
- use existing APIs to query surface at run-time
- open issue: all `<Require>`s apply to all `<Content>` blocks, is this acceptable?

0.7: miscellaneous

- built-in JSON support
- clarification of requestPermission and batching
- clarification of default data ACLs

Deferred to 0.8+

0.8 and beyond

- types for data API
- built-in support for "top friends"
- UI templating -- extension of activity stream template syntax
- APIs for UI components, e.g. friend picker, tabs, etc.
- linking multiple gadgets as part of the same application
- image caching and spriting
- analytics
- per-key ACLs in data API
- standard for "deep linking" on canvas pages, shortnames

Demo/Code

From the OpenSocial Tutorial

http://code.google.com/apis/opensocial/articles/tutorial/

Caja Javascript sanitizer

- Open Source project from Google
- Optional but recommended for OpenSocial containers
- Gadgets can be a new vector for phishing, spam, malware
- Social spread of gadgets can spread bad gadgets too
- Caja reduces threats with a JavaScript sanitizer as an additional "sandbox" on top of iFrames protection
- The Caja sandbox/sanitizer will eventually be secure enough to run gadgets inline instead of in iframes to improve performance

Who should care about Caja?

- OpenSocial Containers can choose to require Caja's extra security, and decide how to handle gadgets that cannot be Cajoled. For example, you might show users an additional warning before they add a non-Cajoled gadget. Of course, you have the option to not use Caja at all.
- Developers writing OpenSocial gadgets may find that some large containers require Cajoled gadgets or warn users of uncajolable gadgets.
- Users, containers and developers will get benefits from Cajoled gadgets running inline (without iFrames) in the long term

Shindig: What is it?

- Apache Software Foundation project
  - Brian McCallister from Ning championed
- Open source reference implementation of OpenSocial + Gadgets stack
- Goal:
  - Launch a new (simple) container in under an hour's worth of work

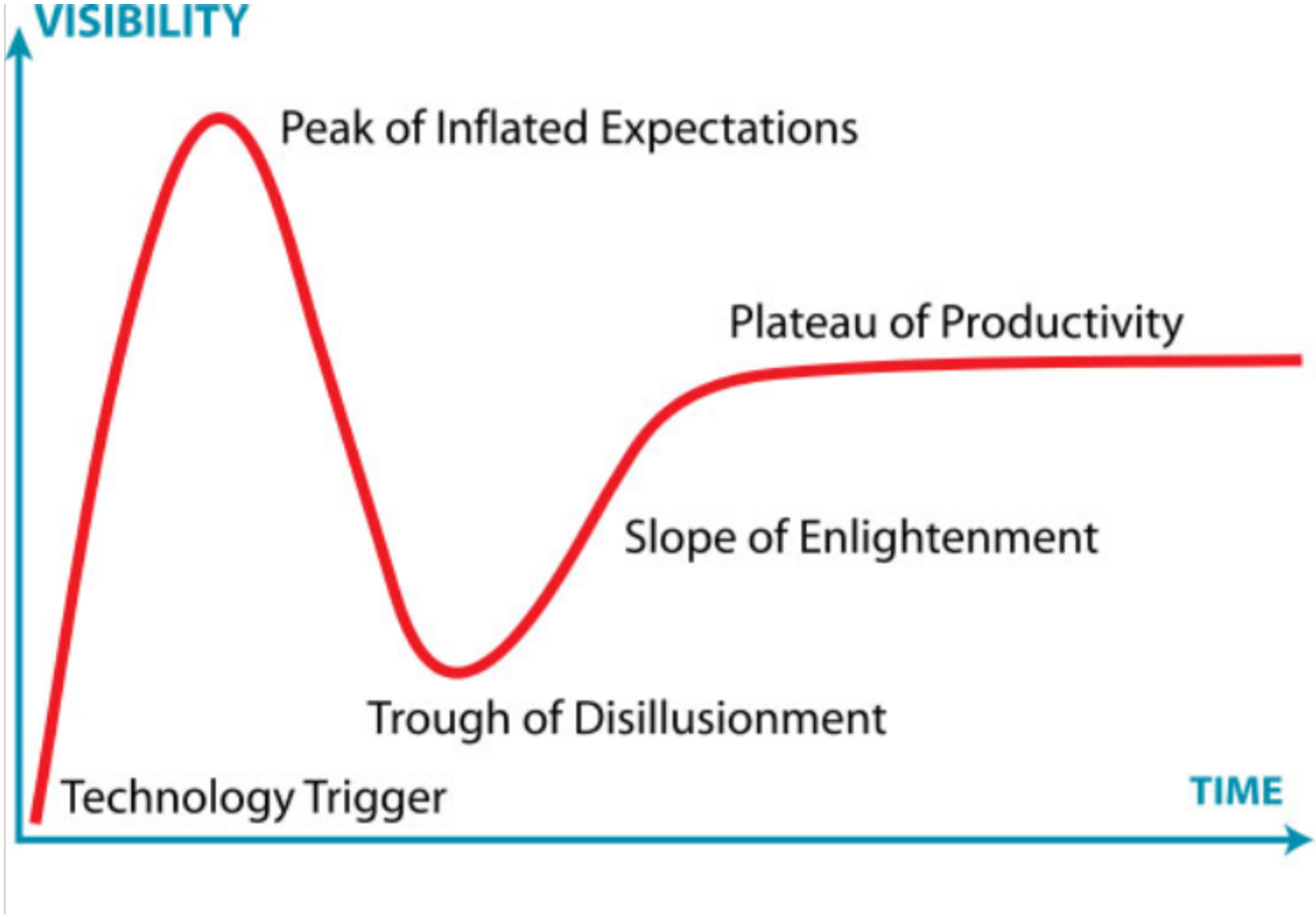
Shindig: Components

- Gadget Container JavaScript
  - Core gadgets JavaScript environment
- Gadget Server
  - Renders gadget XML (i.e. gmodules.com)
- OpenSocial Container JavaScript
  - JavaScript environment for people, activities, persistence
- OpenSocial Gateway Server
  - RESTful API server

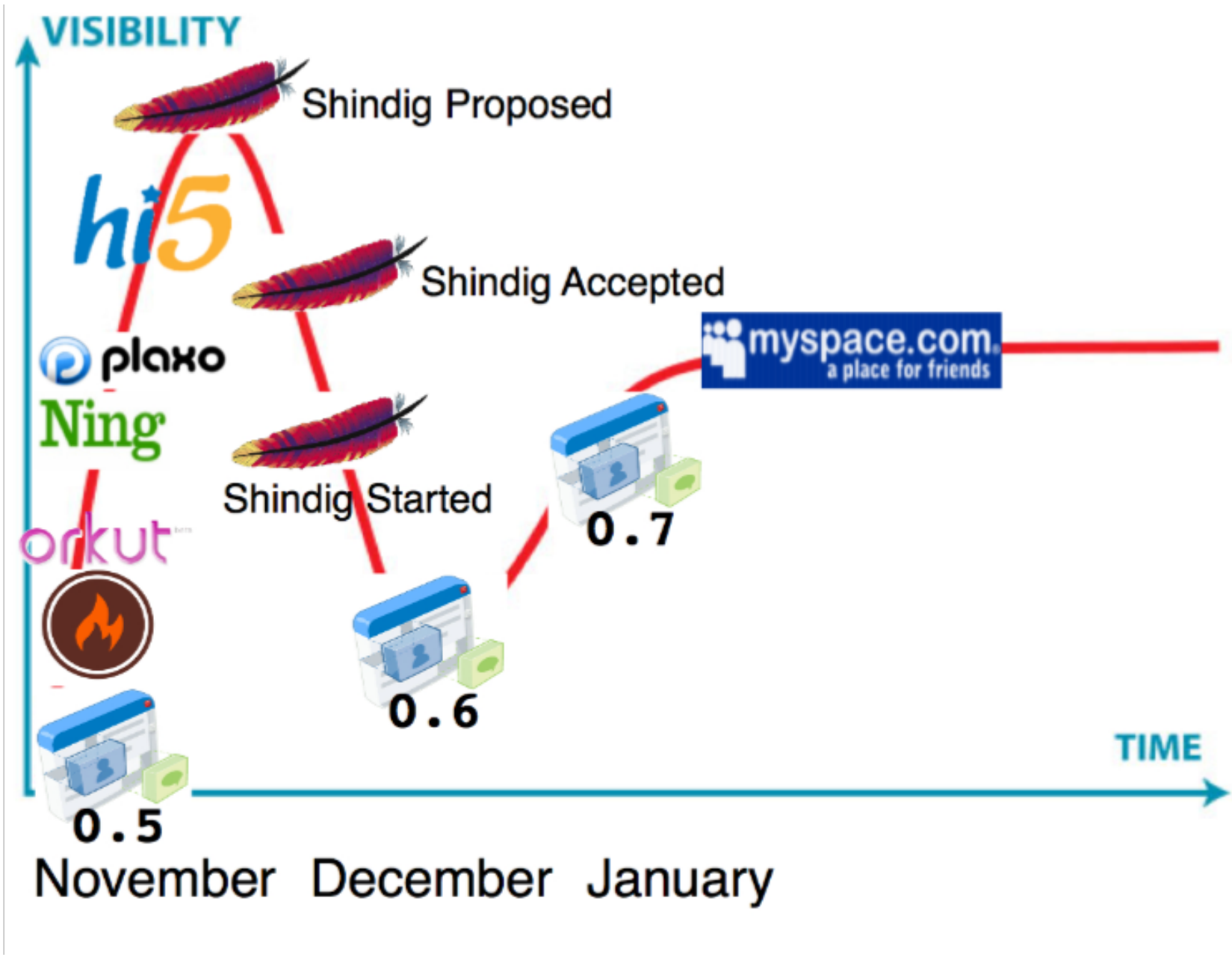
Code Status

- Not finished: javascript not plugged to java layer yet
- Googlers have contributed:
  - Java Gadget Server
  - Gadget Container JavaScript
  - OpenSocial Container JavaScript
- PHP Gadget Server contributed by Ning
- Rumors of Ruby, Python, Perl, C#, etc.

Gartner Technology Hype Cycle



OpenSocial Hype Cycle



Conclusion: it's time to get productive!

OpenSocial is a standard for social applications

The current version 0.7 is the version that will go in production

Developers can start creating social applications today

- Orkut sandbox, Hi5, Plaxo, Ning (and Feb 5 MySpace)

First containers to open for consumers in Q1

Social sites who want to implement OpenSocial should look at Shindig and start planning

Resources For Application Developers

Specification
http://code.google.com/apis/opensocial/

Code Samples and Tools
http://code.google.com/p/opensocial-resources/

Sandboxes
http://developer.myspace.com/
http://www.hi5networks.com/developer/
http://opensocial.ning.com/
http://pulse.plaxo.com/pulse/gadgets/
http://code.google.com/apis/orkut/

My delicious feed: http://del.icio.us/chanezon/opensocial

Resources For Container Developers

Specification
http://code.google.com/apis/opensocial/

For container developers
http://incubator.apache.org/shindig/
http://code.google.com/p/google-caja

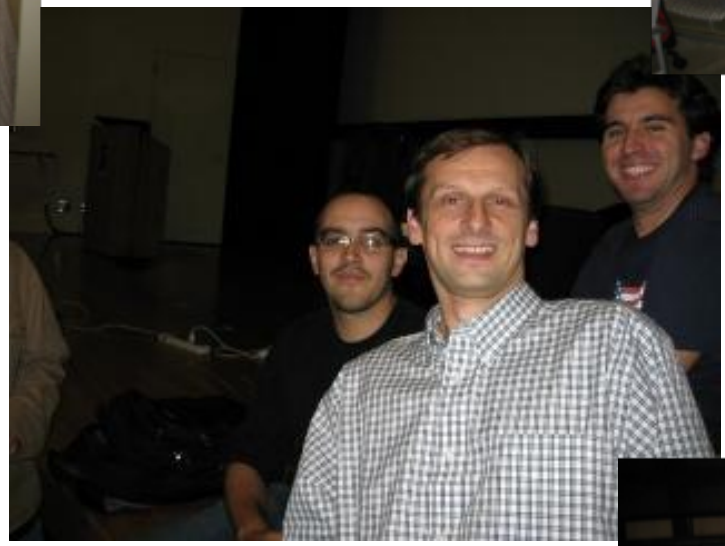
My delicious feed: http://del.icio.us/chanezon/opensocial

Creating OpenSocial is Social


Guest Wifi

TALKING

LAUGHING




WE
HELP
EACH
OTHER!

WE
READ
TOGETHER!

WE
DO PROJECTS
TOGETHER.

BUT WHEN WE ARE MAD...

IT NEVER FINISH WELL...

SO
WE
STOP
FIGHTING!

A standard for everyone

Questions