Roller and blogs as a web development platform

Dave Johnson
Sun Microsystems, Inc.

ApacheCon
leading the wave of open source

May 1-4 2007

Session Goals: to learn...

Ways that blogs, planets, feeds, wikis and can be integrated and automated

How they enable the self-service web for users of varying skill levels

How Apache Roller can be customized and extended via templates, scripting and plugins

Agenda

- Overview
- Blog, wiki and feed Web services
- Blog customization via templates
- Blog server customization via plugins
- Example: Scripting languages in Roller
- Planet server aggregation
- Use case: Software business blogs

The self-service web

- chmod 777 web
- Let everybody write the web

- Get IT and web-masters out of the way
- Give the tools to the users
  - Blogs, wikis and feeds

The self-service web

- Blogs, wiki and feed technologies
- Make it easy for your users to:
  - Read, write and monitor the web
  - To build interesting sites & seed communities
  - Create useful information feeds
- With scripts, templates
- Or no code at all

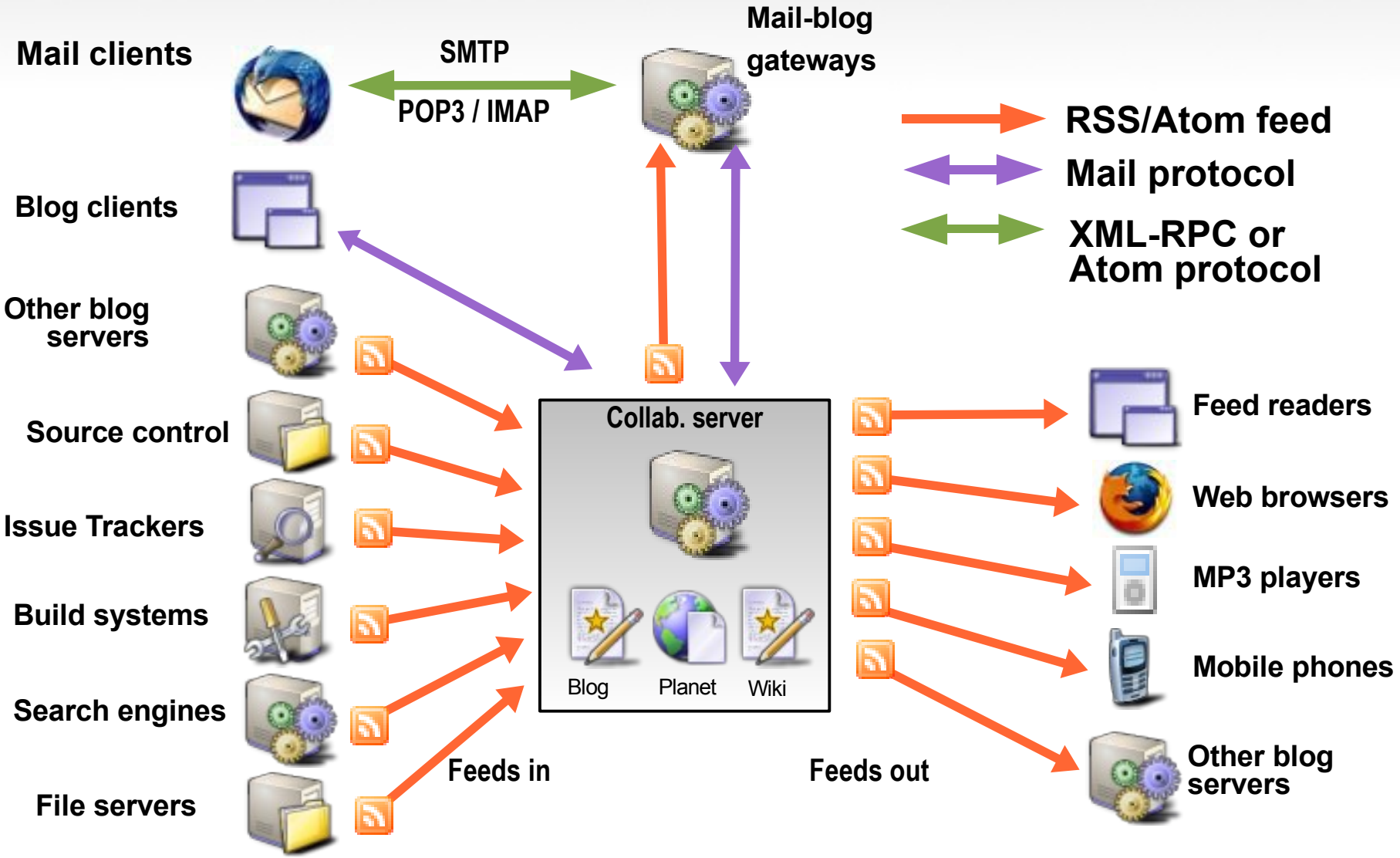
With no code at all

- Easy web publishing and feed generation
  - Easy interface for writers
  - Feeds for entries and comments
  - Filtered feeds by category, tags and search
  - Aggregated feeds
  - Podcast / media feeds

Or with a little code...

- With a little know-how
  - Custom blog pages and feeds
  - Feed aggregations
- With a little help from the IT department
  - Custom publishing clients
  - Weblog entry plugins
  - Weblog rendering plugins

Feed-based integration



Agenda

- Overview
- **Blog, wiki and feed Web services**
- Blog customization via templates
- Blog server customization via plugins
- Example: Scripting languages in Roller
- Planet server aggregation
- Use case: Software business blogs

Blog, wiki and feed web services

- Blogger API / MetaWeblog API
- Atom Publishing Protocol

- Roller Admin Protocol

- WikiRPCInterface

Meta Weblog API

- Extends super-simple Blogger API
- Adds RSS-based metadata
- Supports
  - Multiple Blogs per User
  - Posting, editing and deleting Blog Posts
  - Get X recent Blog Posts, but no paging
  - Uploading binary files

MetaWeblog API: XMLRPC methods

getUserBlogs	Get blogs as array of structures
newPost	Create new blog post by passing in structure
getPost	Get blog post by id
getRecentPosts	Get most recent N blog posts
editPost	Update existing blog post
deletePost	Delete blog post specified by id
newMediaObject	Upload file to blog (e.g. picture of my cat)
getCategories	Get categories allowed in blog

MetaWeblog API: Java example

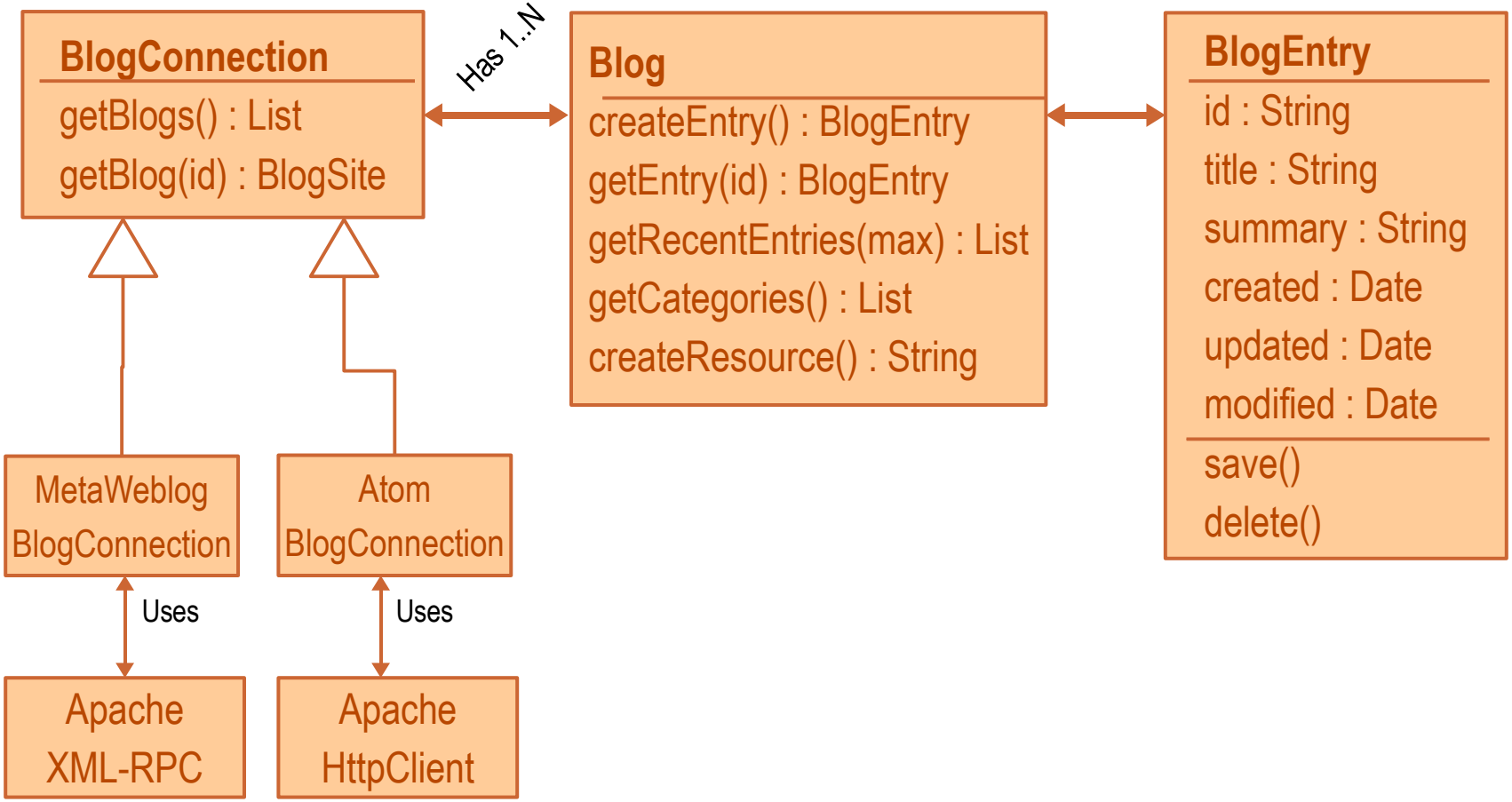
```
String uname, pword, blogid, endpoint, title, content;

Hashtable post = new Hashtable();
post.put("title", title);
post.put("description", content);

XmlRpcClientConfigImpl xmlrpcConfig =
   new XmlRpcClientConfigImpl();
xmlrpcConfig.setServerURL(new URL(endpoint));
XmlRpcClient client = new XmlRpcClient();
client.setConfig(xmlrpcConfig);

Integer result = (Integer)client.execute("metaWeblog.newPost",
   new Object[] {blogid, uname, pword, post, Boolean.TRUE});
```

ROME Propono Blog Client



ROME Propono Blog Client example

```
BlogConnection conn =
   BlogConnectionFactory.getBlogConnection(
      "metaweblog", endpoint, username, password);

// get first blog available to user
Blog blog = (Blog)conn.getBlogs().get(0);

// create and post new blog entry
BlogEntry entry = blog.newEntry();
entry.setTitle("NPE on submitting new order query");
entry.setContent(
  new BlogEntry.Content("This is a <b>bad</b> one!"));
entry.save();
```

Atom Publising Protocol (APP)

- Everything MetaWeblog API does
  - Atom does better
- REST based publishing protocol
- Create, edit, update & retrieve collections
  - Blog entries, podcasts, uploads, etc.
  - Wiki pages, attachments, etc.
  - Bookmarks, photos, calendar events, etc.
  - All that is bloggy...

Atom Protocol: Service Doc

```
<service xmlns="http://purl.org/atom/app#">
    <workspace title="My Weblog" >

        <collection title="Weblog Entries"
            href="http://hostname/app/myweblog/entries">
            <accept>entry</accept>
        </collection>

        <collection title="Uploaded Files"
            href="http://hostname/app/myweblog/resources">
            <accept>image/*</accept>
        </collection>

    </workspace>
</service>
```

Atom collection <feed>

```
<feed xmlns="http://www.w3.org/2005/Atom">

  <link rel="next"
    href="http://hostname/app/myweblog/entries/60"/>

  <link rel="previous"
    href="http://hostname/app/myweblog/entries/30"/>
  ...
  <entry> ... </entry>
  <entry> ... </entry>
  <entry> ... </entry>
  <entry> ... </entry>
  ...
</feed>
```

URIs for next and previous portions of collection

<entry> in a collection

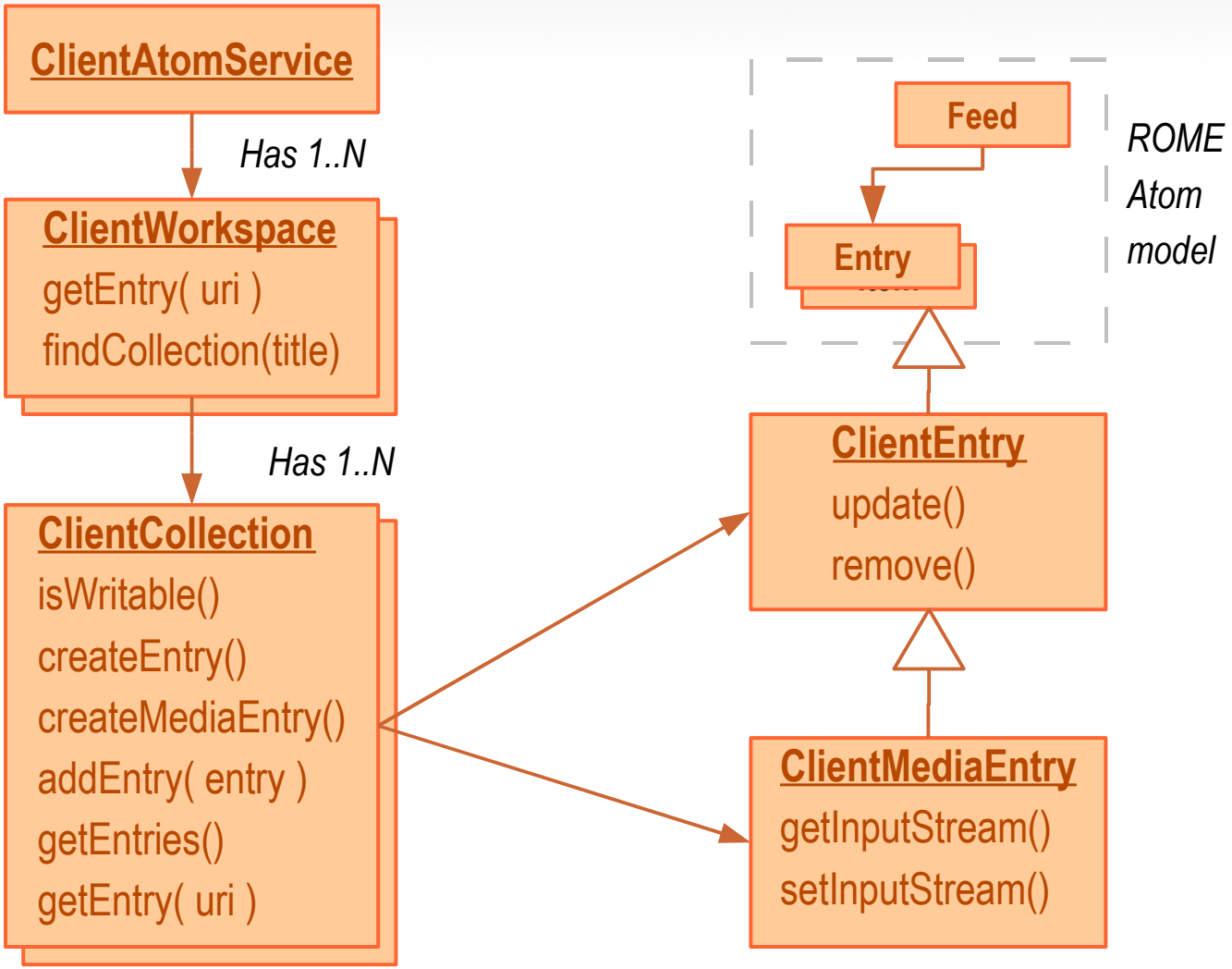
```
<entry>
  <title>Unit test results</title>
  <link rel="alternate"
    href="http://localhost/roller/page/bill?entry=post1 />
  <link rel="edit"
    href="http://localhost/roller/app/bill/entry/757" />
  <category term="/UnitTests" />
  <id>http://buildserver/roller/status/entry/report1</id>
  <updated>2005-12-27T22:08:03Z</updated>
  <published>2004-10-13T01:07:59Z</published>
  <content type="html">Content goes here...</content>
  <app:control>
    <app:draft>no</app:draft>
  </app:control>
</entry>
```

Entry's permalink

Edit URI for entry

Draft status

ROME Propono Atom Client



ROME Propono Atom Client example

```
ClientAtomService service =
   AtomClientFactory.getAtomService(endpoint, uname, pword);

// find workspace for Order System project
ClientWorkspace workspace =
   (ClientWorkspace)service.findWorkspace("Order System");

// find first collection that accepts entries
ClientCollection collection =
   (ClientCollection)ws.findCollection(null, "entry");

ClientEntry entry = collection.createEntry();
entry.setTitle("NPE on submitting new order query");
entry.setContent(Content.HTML, "This is a <b>bad</b> one!");
collection.addEntry(entry);
```

Roller Admin Protocol

- REST based protocol for Roller
  - Add, edit and delete Users
  - Add, edit and delete Weblogs
  - Add, edit and delete Members

- Some similarities to Atom protocol
  - IMHO: Replace with APP

Roller Admin Protocol: Service Doc

```
<service xmlns="http://purl.org/roller/rap#">
  <workspace title="Collections for administration">
    <collection title="Weblog administration entries"
      href="http://suntoy:8080/roller/rap/weblogs">
      <member-type>weblog</member-type>
    </collection>
    <collection title="User administration entries"
      href="http://suntoy:8080/roller/rap/users">
      <member-type>user</member-type>
    </collection>
    <collection title="Member administration entries"
      href="http://suntoy:8080/roller/rap/members">
      <member-type>member</member-type>
    </collection>
  </workspace>
</service>
```

Roller Admin Protocol: User Set

```
<users xmlns="http://purl.org/roller/rap#"
   href="http://example.org/roller/rap/users">

  <user href="http://example.org/roller/rap/users/mort">
    <name>mort</name>
    <full-name>Mortimer Snerd</full-name>
    <password>edgar</password>
    <locale>en</locale>
    <timezone>US/Pacific</timezone>
    <email-address>mort.snerd@example.org</email-address>
    <date-created>1141863448000</date-created>
  </user>

</users>
```

WikiRPCInterface: XMLRPC methods

getRecentChanges	Get recent changes as array
getPage	Get the raw Wiki text of page, latest version
getPageVersion	Get the raw Wiki text of page, by version
getPageHTML	Return page in rendered HTML, latest version
getPageHTMLVersion	Return page in rendered HTML, by version
getPageInfo	Get page info as struct, latest version
getPageInfoVersion	Get page info as struct, by version
getAllPages	Returns a list of all pages

WikiRPCInterface

- Supported by:
  - MoinMoin, Twiki, JSPWiki, PHPWiki
  - But not Confluence, which has it's own API
- Limitations:
  - Read-only, no paging
- Future
  - WikiRPCInterface2 vs. Atom protocol

Agenda

- Overview
- Blog, wiki and feed Web services
- **Blog customization via templates**
- Blog server customization via plugins
- Example: Scripting languages in Roller
- Planet server aggregation
- Use case: Software business blogs

Roller Weblog Templates

- A weblog has three required templates
  - Weblog, _day and _css
- Plus any number of user defined templates

- Template language: Apache Velocity
  - Other languages possible via plugins

- Data available to templates via *Models*

Roller Model Objects

PageModel

locale
tags
weblog
weblogCategory
weblogEntriesPager
weblogEntry
isPermalink
...

UtilitiesModel

decode()
encode()
escapeHTML()
escapeJavaScript()
escapeHTML()
formatDate()
formatISO8601Date()
...

URLModel

home()
feed()
login()
logout()
page()
resource()
search()
...

MenuModel

PreviewModel

MessageModel

CalendarModel

SearchResultsFeedModel

Pager

homeLink
homeName
nextLink
nextName
prevLink
prevName
items

ConfigModel

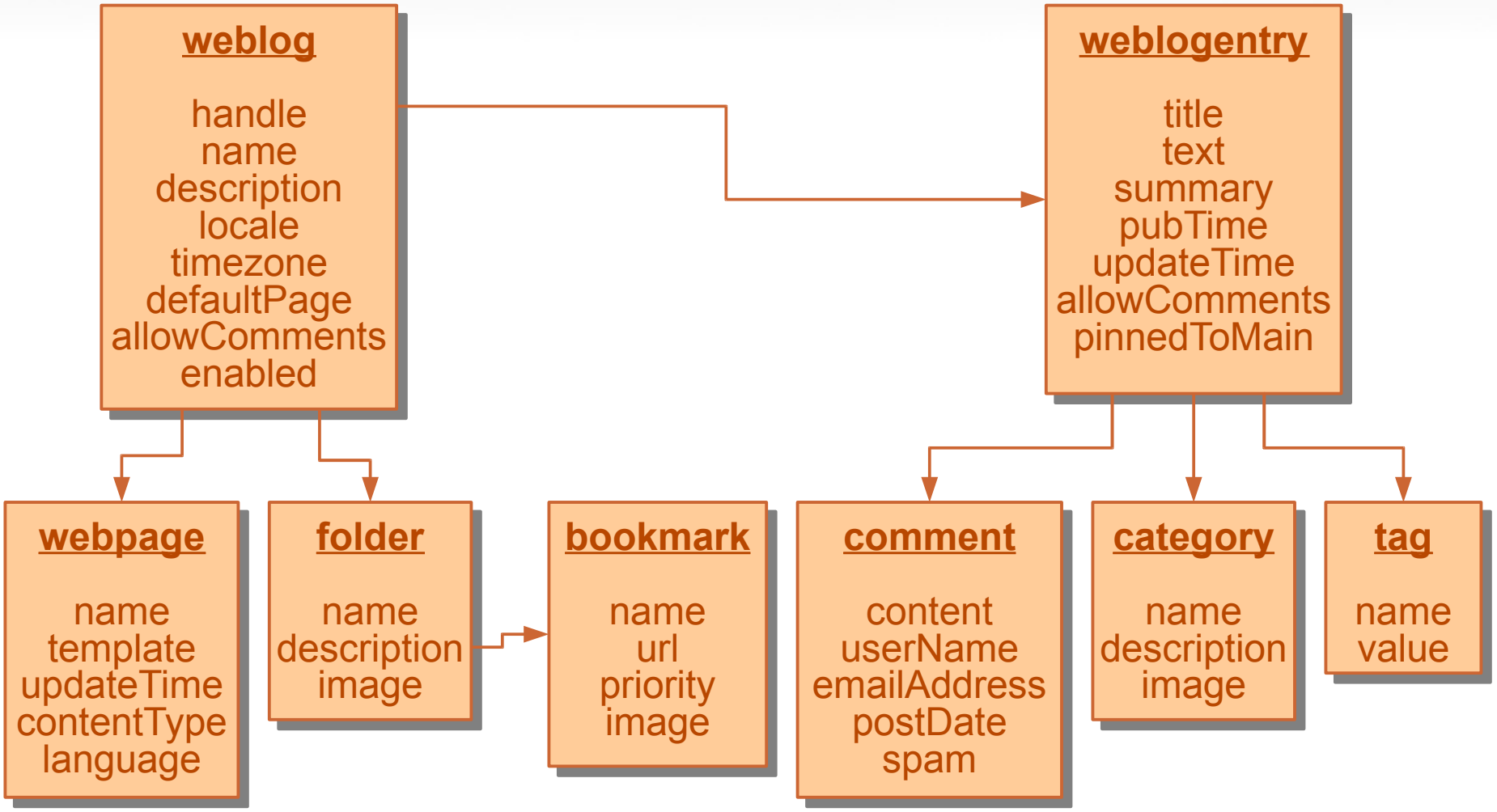
SearchResultsModel

PlanetModel

FeedModel

SiteModel

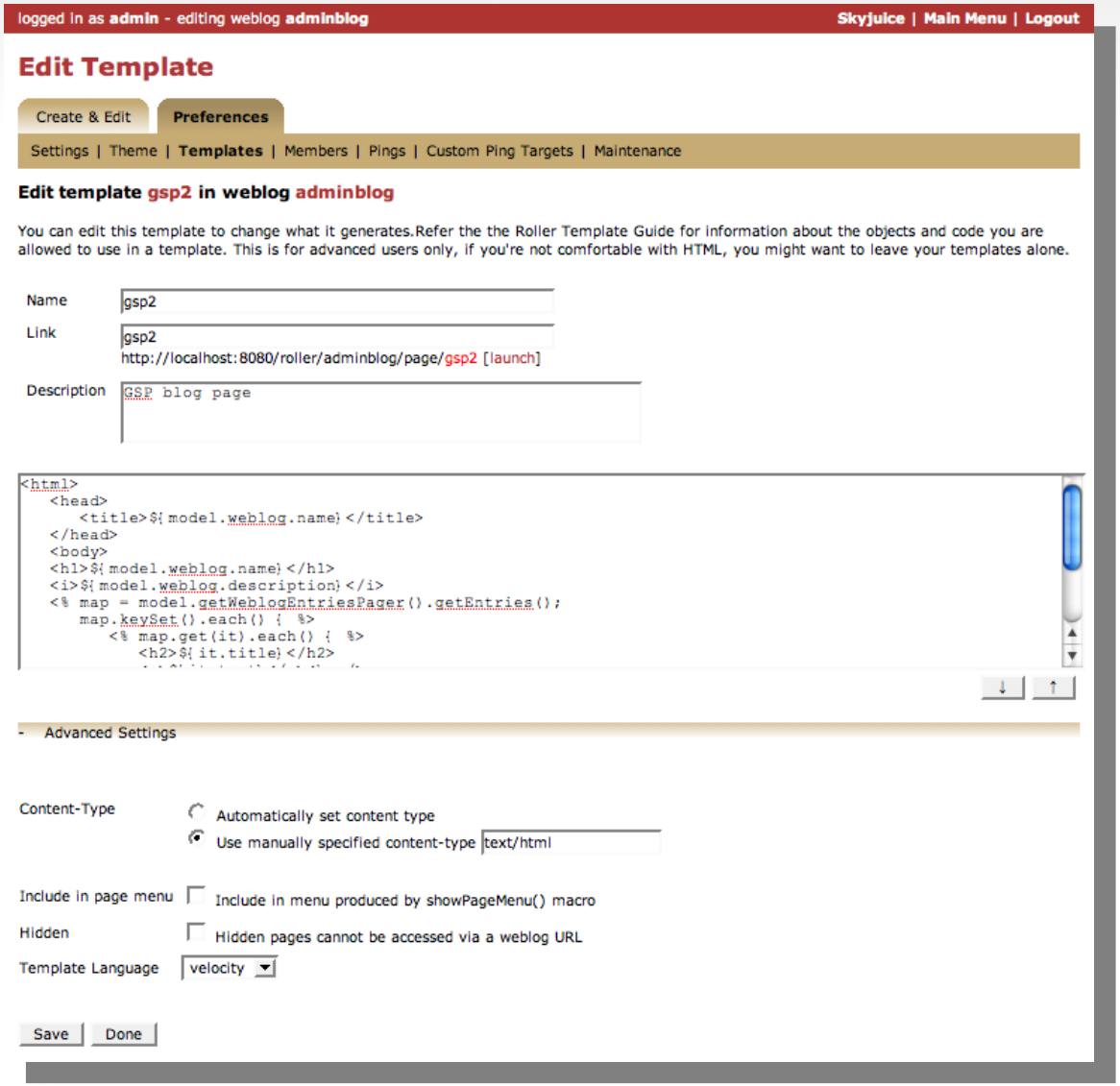
Roller Data Model



Why customize templates?

- Add new pages and feeds to your blog
  - HTML, XHTML pages
  - JSON pages for Ajax effects
  - Custom RSS and Atom feeds
- Using data from
  - Roller's models and POJOs
  - Feed data from built-in Planet
  - Custom model objects

Roller Template Editor



Example: Template for JSON

```
#set($pager = $model.getWeblogEntriesPager()) {
#if($pager.nextLink) nextLink:"$pager.nextLink", #end
#if($pager.prevLink) prevLink:"$pager.prevLink", #end

data: [
#foreach($day in $pager.getEntries().keySet())
   #set($entries = $pager.getEntries().get($day))
   #foreach($entry in $entries)
      #if($comma),#else#set($comma = true)#end
      {id: "$entry.website.handle:$entry.anchor",
       pubTime: "$entry.pubTime", title:"$entry.title"}
   #end
#end ]
}
```

Example: JSON template output

```
{ nextLink:"/roller/page/dojoentries?page=1",
data: [
  { id:      "roller:latest_links14",
    pubTime: "2007-03-21 22:42:13.0",
    title:   "Latest links: March 21, 2007" }
  { id:      "roller:jruby_on_roller",
    pubTime: "2007-03-20 21:46:23.0",
    title:   "JRuby on Roller" }
  { id:      "roller:congrats1",
    pubTime: "2007-03-20 13:02:07.0",
    title:   "Congrats" }
  { id:      "roller:roller_on_sjs_web_server",
    pubTime: "2007-03-20 13:02:02.0",
    title:   "Roller on SJS Web Server 7.0" }
  ...
] }
```

Example: Dojo table in a Roller page

Agenda

- Overview
- Blog, wiki and feed Web services
- Blog customization via templates
- **Blog server customization via plugins**
- Example: Scripting languages in Roller
- Planet server aggregation
- Use case: Software business blogs

What can you do with Roller plugins?

- Transform entry content at display time
- Plugin your own models for use in templates
- Plugin your own template language
- Add your own comment validation rules

Which parts of Roller API to use

- Roller Manager Interfaces
  - UserManager, WeblogManager, etc.
- Roller POJOs
  - WeblogEntryData, CommentData, etc.
- Roller Configuration
  - RollerConfig, RollerRuntimeConfig
- Roller Utilities
- Request object(s)

Roller Weblog Entry Plugins

- Transform entries at display time
- Implement this interface

```
public interface WeblogEntryPlugin {
   public String getName();
   public String getDescription();
   public void init(WebsiteData weblog)
       throws RollerException;
   public String render(
       WeblogEntryData entry, String s);
}
```

- Add class to config property:

```
plugins.page=
```

Weblog Entry Plugin examples

- JSPWiki plugin
- Emoticon plugin
- Technorati Tag plugin
- Wikipedia plugin
- Google plugin
- Textile plugin

Roller Page Model Plugins

- Plugin new models for use in templates
- Implement model interface

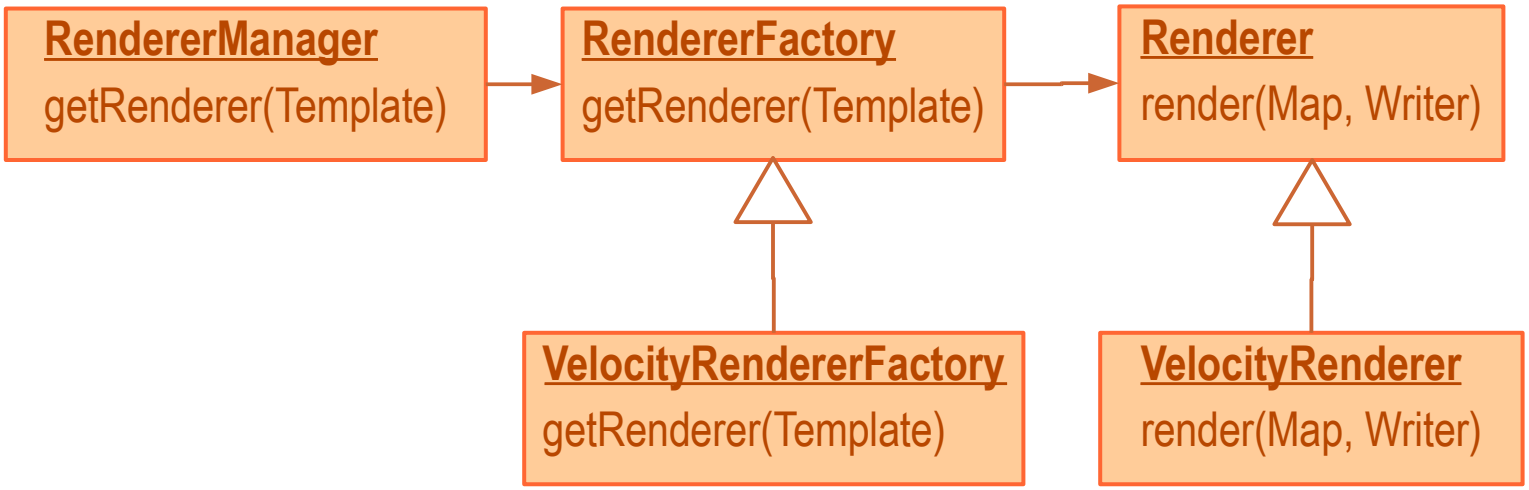
```
public interface Model {
    public String getModelName();
    public void init(Map params);  // called each request
}
```

- Add to appropriate configuration properties

```
rendering.pageModels=       // for weblog templates add
rendering.searchModels=     // your plugin to these three
rendering.previewModels=    // lists of model classes

rendering.feedModels=       // for built-in feeds
```

Roller Page Rendering



Roller Renderer Plugins

- Allows you to plugin your own view
- Implement this factory interface:

```
public interface RendererFactory {
   public Renderer getRenderer(Template template);
}
```

- Return null if you don't want to render
- And this render interface:

```
public interface Renderer {
   public void render(Map model, Writer writer)
      throws RenderingException;
}
```

Comment Validation Plugins

- Plugin your own comment validation
- Implement this interface:

```
public interface CommentValidator {
   public String getName();
   public int validate(
      CommentData comment, RollerMessages messages);
}
```

- Return 0 if spam and 100 if not
- Add your class to the list of validators

```
comment.validator.classnames=
```

Existing Comment Validator Plugins

- ExcessSizeCommentValidator
- ExcessLinksCommentValidator
- AkismetCommentValidator
- BlacklistCommentValidator

Agenda

- Overview
- Blog, wiki and feed Web services
- Blog customization via templates
- Blog server customization via plugins
- **Example: Scripting languages in Roller**
- Planet server aggregation
- Use case: Software business blogs

Scripting in languages in Roller

- Author pages using scripting language
- Bean Scripting Framework (BSF)
  - JRuby, Javascript, Jython Groovy
- Groovy Server Pages (GSP) Renderer
  - JSP style pages w/Groovy

Example: a BSF Renderer

```
public void render(Map model, Writer writer)
//...
    BSFManager manager = new BSFManager();

    for (it = model.keySet().iterator(); it.hasNext();) {
      String key = (String)it.next();
      manager.declareBean(key,
         model.get(key), model.get(key).getClass());
      manager.registerBean(key, model.get(key));
    }
    manager.declareBean("out", writer, Writer.class);
    manager.registerBean("out", writer);

    manager.exec(template.getTemplateLanguage(),
      "(java)", 1, 1, template.getContents());
//...
}
```

JRuby via BSF Renderer

```
$out.println "<html><head>"
$out.println "<title>#{$model.weblog.name}</title>"
$out.println "</head><body>"
$out.println "<h1>#{$model.weblog.name}</h1>"
$model.weblogEntriesPager.entries.keySet().each {
    |day|
    $model.weblogEntriesPager.entries.get(day).each {
        |entry|
        $out.println "<h3>#{entry.title}</h3>"
        $out.println "<p>#{entry.text}</p>"
    }
}
$out.println "</body></html>"
```

Groovy Server Pages (GSP) Renderer

```
<html>
   <head>
      <title>${model.weblog.name}</title>
   </head>
   <body>
   <h1>${model.weblog.name}</h1>
   <i>${model.weblog.description}</i>
   <% map = model.getWeblogEntriesPager().getEntries();
      map.keySet().each() { %>
          <% map.get(it).each() { %>
            <h2>${it.title}</h2>
            <p>${it.text}</p><br />
          <% }
      }%>
   </body>
</html>
```

Planet server aggregation

- Overview
- Blog, wiki and feed Web services
- Blog customization via templates
- Blog server customization via plugins
- Example: Scripting languages in Roller
- **Planet server aggregation**
- Use case: Software business blogs

Planets

PLANET APACHE

March 26, 2007

Dan Diephouse — RESTful reliability continued...

Stefan Tilkov has been doing a good job covering the ongoing discussion of how to do POSTs reliably.

All the solutions for a reliable POST seem ugly.

1. *POST to the `http://example` it return `201 Created` and the header.*
2. *PUT the intended content to the*

While it has beauty in consistency, it seems les addition of a resource reliably.

On the other hand we have WS-RM. The nice t even if it does take a message based approach i WS-RM is XML only. One of the major benefit opinion is that works with any type of resource and REST camps merging together.

Andrew Savory — Wine

Those of my friends, family, and colleagues of

SUBSCRIPTIONS

Adam
Adria
Alex
Andr
Apac
Apac
Apac

GNOME™ About GNOME · Download · **Users** · Art & Themes · Developers · Foundation · Contact

planet mozilla

March 26, 2007

Mike Pinkerton

Must be in my other pants...
Have you ever stood at the register, frantically searching your wallet for your Safeway card, only to realize that you're...

March 26, 2007 12:30 AM

March 25, 2007

Robert Kaiser

Pingback and TrackBack: ease of implementation (or not)
Finally I managed to implement pingback in addition to TrackBack, and it was interesting to implement both, to compare them from a developer's perspective - as both are technologies that enable other blogs to link back blog entries that link them and this way create a permanent connection between two blogs.

One target of pingback is said to be that it should be "implementable with minimal effort", I also read in a few places that it should not attract spam as easily as TrackBack. The latter has been achieved quite nicely, as the pingback client needs to tell the server the source URL containing the original link as well as its target, and the server needs to verify this link to this target actually exists in the source. TrackBack on the other hand just sends the the URL to link back to and needs no verifications, so strictly according to the spec, a

Feeds: RSS 2.0, RSS 1.0
Subscription list: FOAF, OPML

Last update: March 26, 2007 03:06 AM
All times are UTC.

Gecko Tree Schedule

Note: These dates are scheduling estimates, and as in most software projects are subject to change. The current milestone is firm, the rest are preliminary and should be used for general guidance only.

Gecko 1.9 alpha 3 Freeze
now
Gecko 1.9 alpha 3
now

Subscriptions

- Adam Sacarny (feed)
- Alan Starr (feed)
- Alex Faaborg (feed)
- Alex Fritze (feed)
- Alex Polvi (feed)
- Alex Vincent (feed)
- Ali Ebrahim (feed)
- AllPeers (feed)
- Allan Beaufour (feed)
- Andy Edmonds (feed)
- Axel Hecht (feed)
- Bart Decrem (feed)
- Ben Goodger (feed)
- Benjamin Smedberg (feed)
- Blake Ross (feed)

Planet GNOME

Planet GNOME

Planet GNOME is a window into the world, work and lives of GNOME hackers and contributors.

Updated on March 26, 2007 03:28 AM UTC. Entries are normalised to UTC time.

Search

Customised search for Planet GNOME contributor blogs.

Google™ Custom Search
Search

GNOME World Wide

GNOME communities around the world by region and language:

What's a Planet?

- A “community aggregator”
- Portal-like web application which displays weblog posts from a group of closely related but separately hosted weblogs
- Provides aggregated feeds so that readers may subscribe to the group as a whole
- Most popular?
  - Planet (Python)
  - See also: Sam Ruby's Venus fork of Planet

Roller-Planet

- Roller's built-in Planet server
  - Used on blogs.sun.com
- Standalone Planet server in development
  - Multiple Planets per user, and users per Planet
  - Each Planet defines Aggregation Groups
  - Based on Roller:
    - Back-end architecture
    - Rendering system

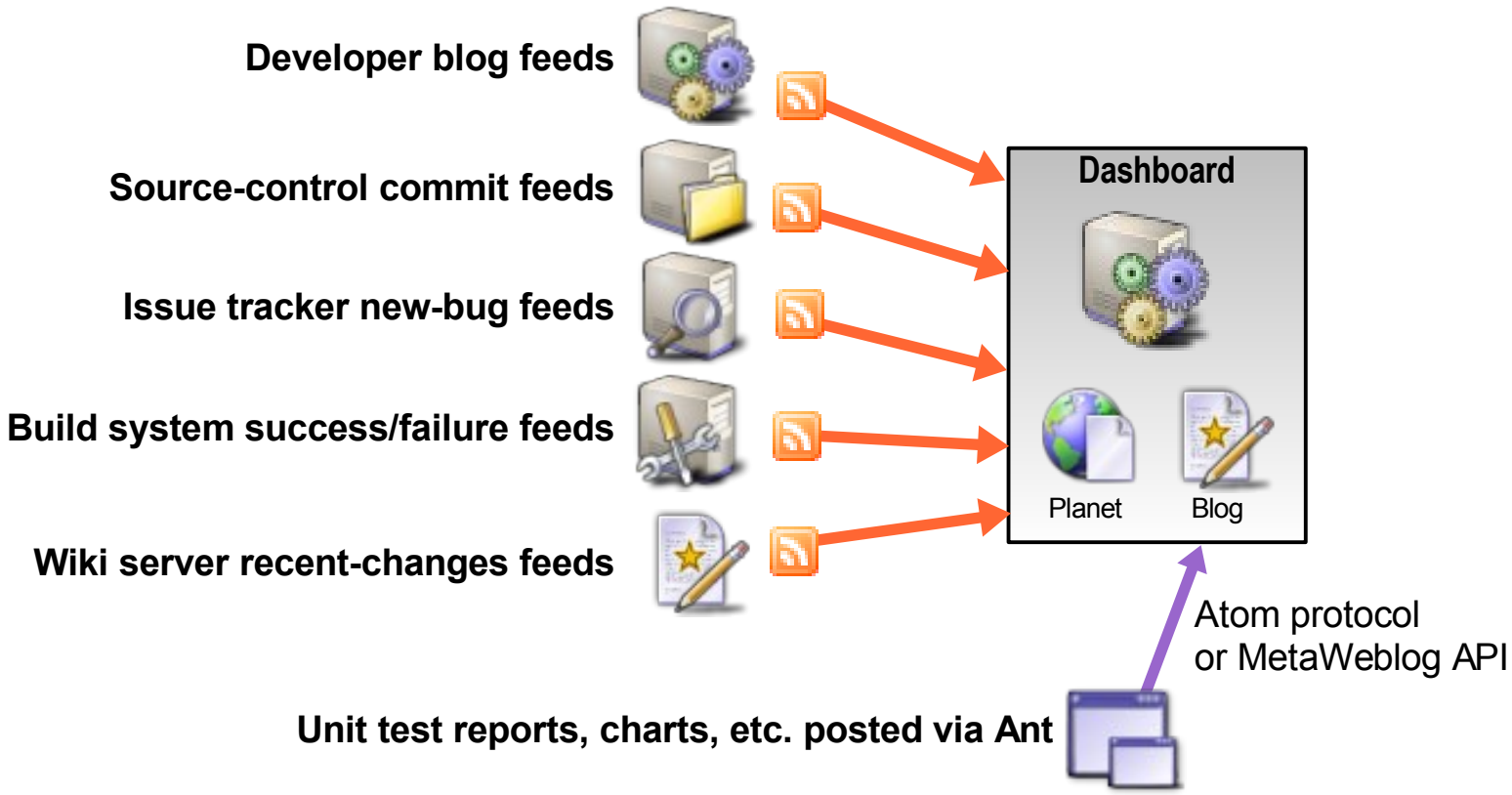
Agenda

- Overview
- Blog, wiki and feed Web services
- Blog customization via templates
- Blog server customization via plugins
- Example: Scripting languages in Roller
- Planet server aggregation
- **Use case: Software business blogs**

Blogs in development

- Development dashboard
- Inputs
  - Feeds from developer systems
  - Test results published by Ant
- Outputs
  - Planet site for developer blogs
  - Variety of aggregated & filtered feeds

Development Dashboard



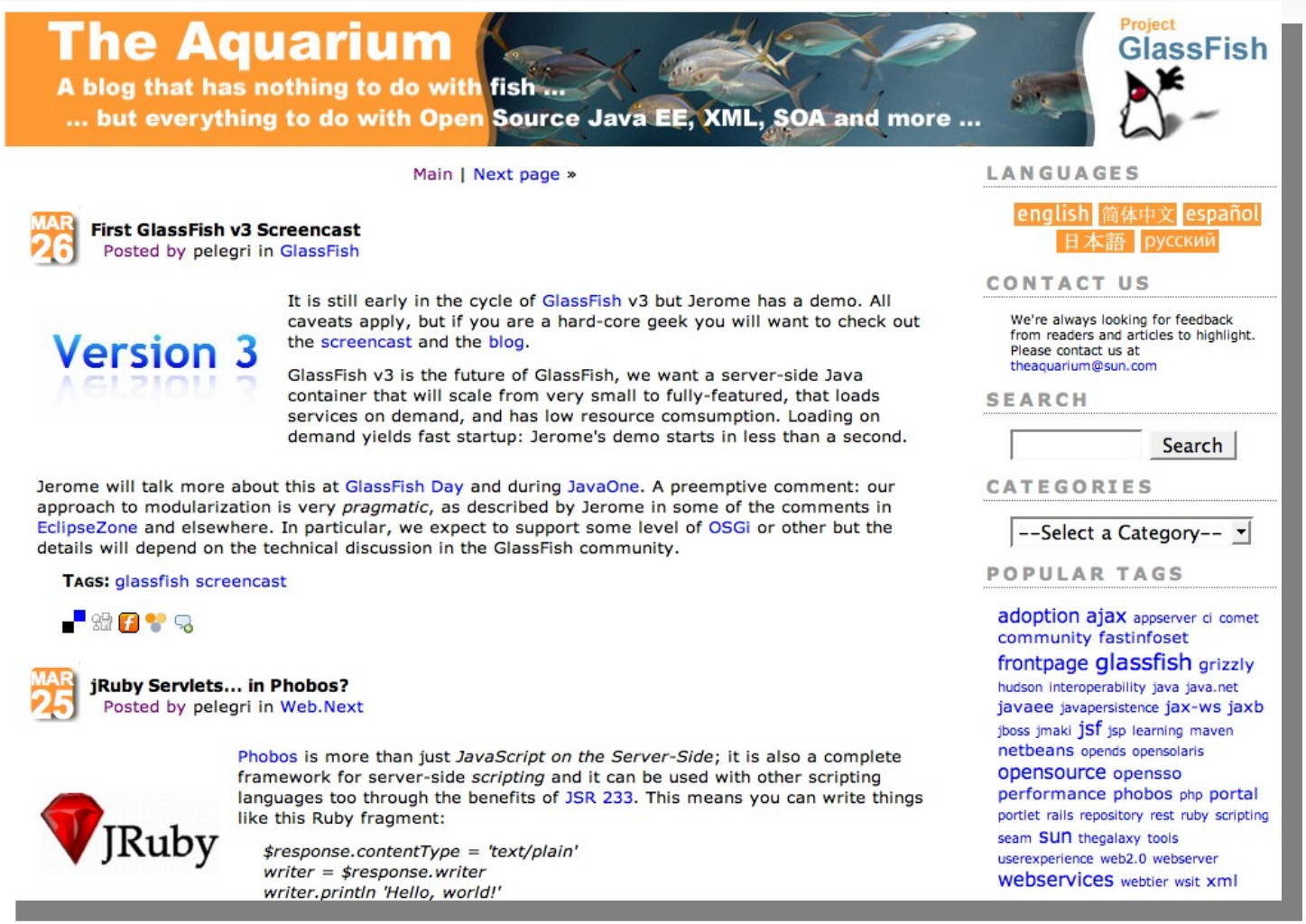
Roller project's dev-dashboard

Blogs in marketing

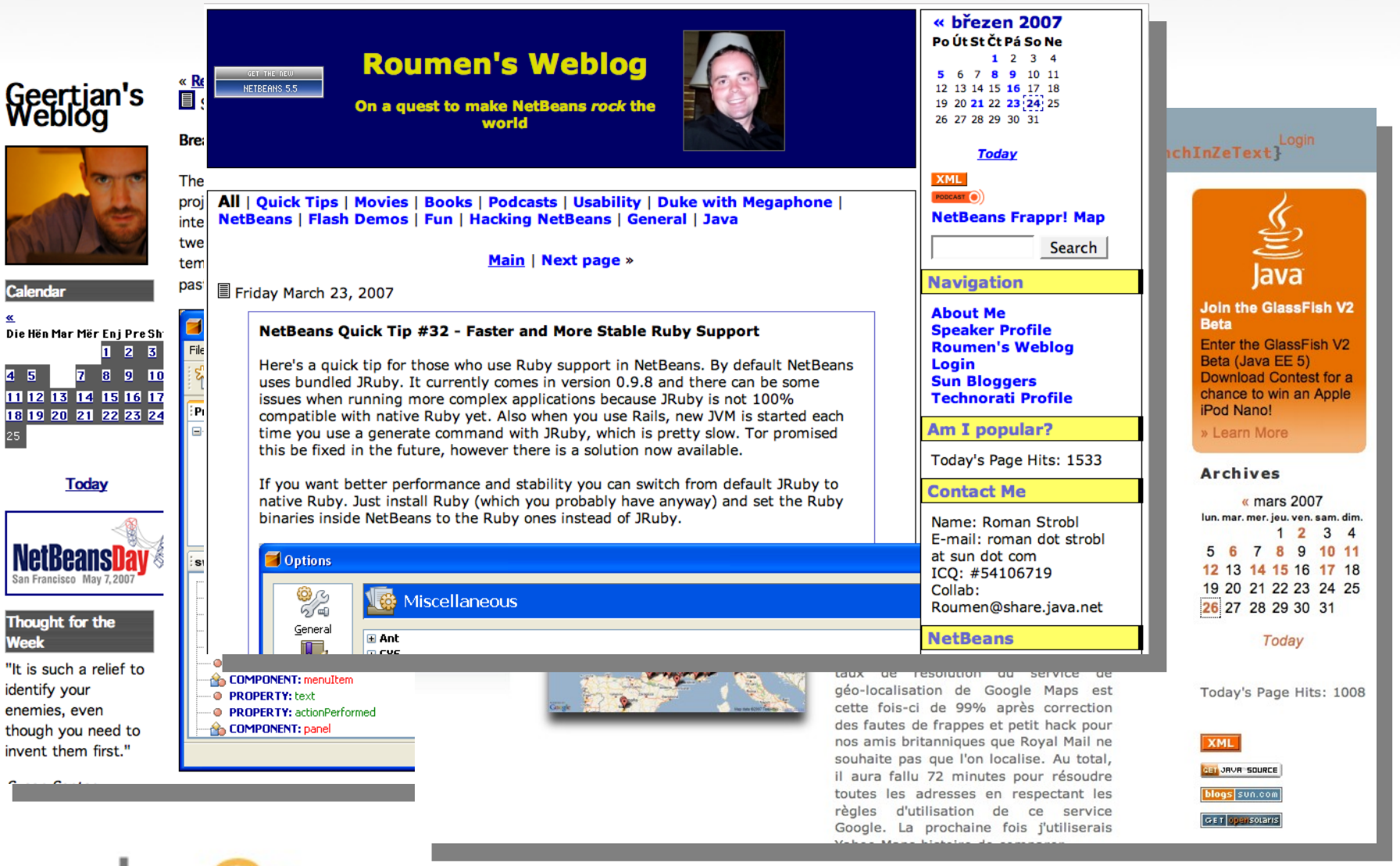
- Product news & links blog
- Developer & evangelist blogs
- Employee blogs via aggregation
- User blogs via keyword blog-search
- Tag based feeds for the online-store

- Look ma, no web master

Product news & links blogs

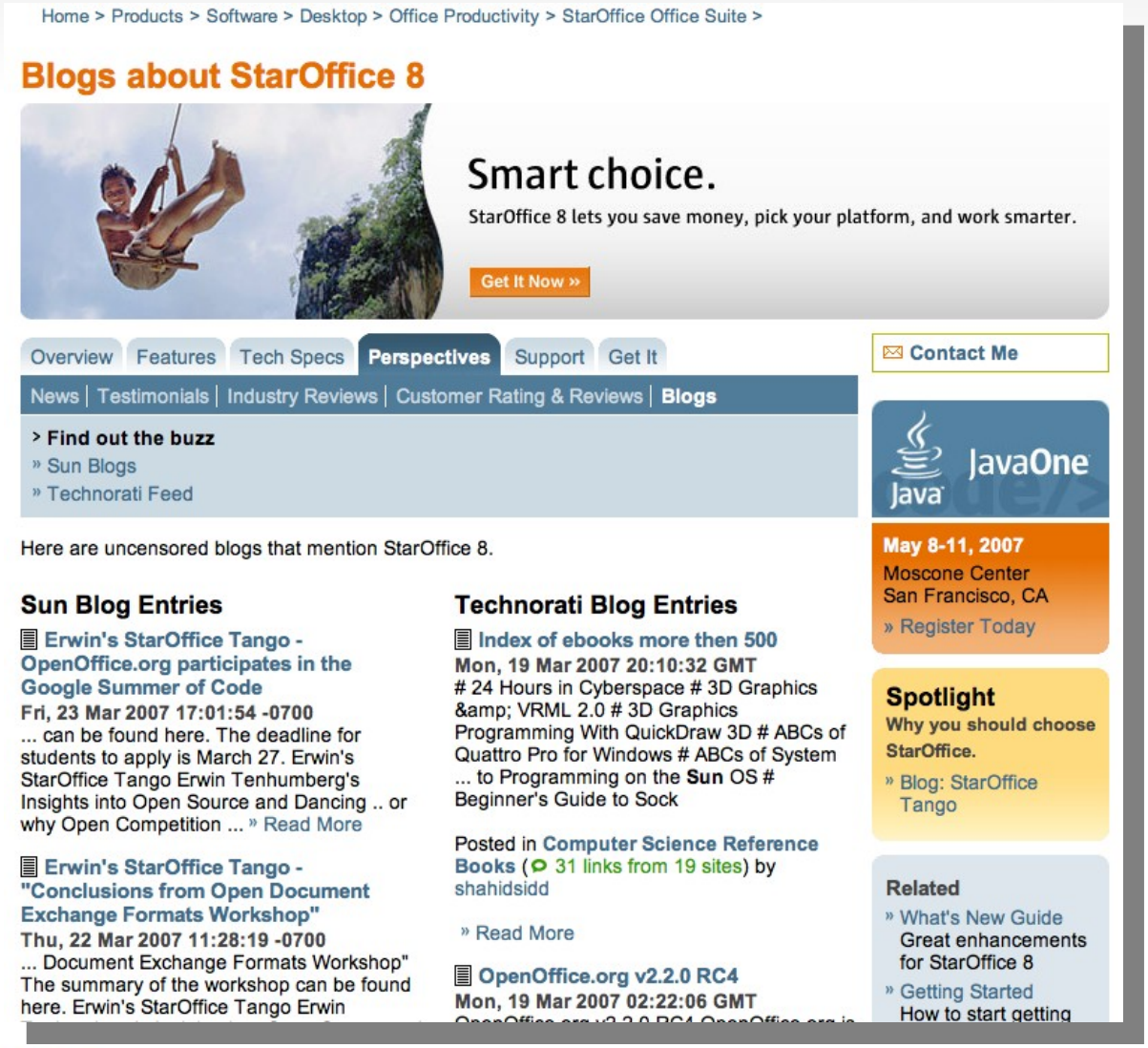


Developer & evangelist blogs



ApacheCon
leading the wave of open source
May 1-4 2007

Blogs in the on-line store



For more information

- Project ROME and Propono
  - http://rome.dev.java.net
- Apache Roller
  - http://roller.apache.org
  - http://roller.dev.java.net (plugins & themes)
  - User Guide & Template Guide
  - Developer Guide (rewrite coming soon)
- Apache Abdera (Incubating)

Summary

- Put blogs, planets, wikis and feeds in place
- Support and encourage their use
- Trust and empower your users/employees
- Enable the self-service web

- Help out with development of
  - Customizations & plugins
  - Better tools via via ASF and F/OSS

Q & A

- Questions?