Interactive Websites: Comet and DWR

Joe Walker, SitePen

sitepen

Agenda:

Comet 101

The Ultimate Hack

Passing the Pain

DWR 101

Demos

sitepen

What is Comet?

AJAX

Comet

What is Comet?

Long lived HTTP connections

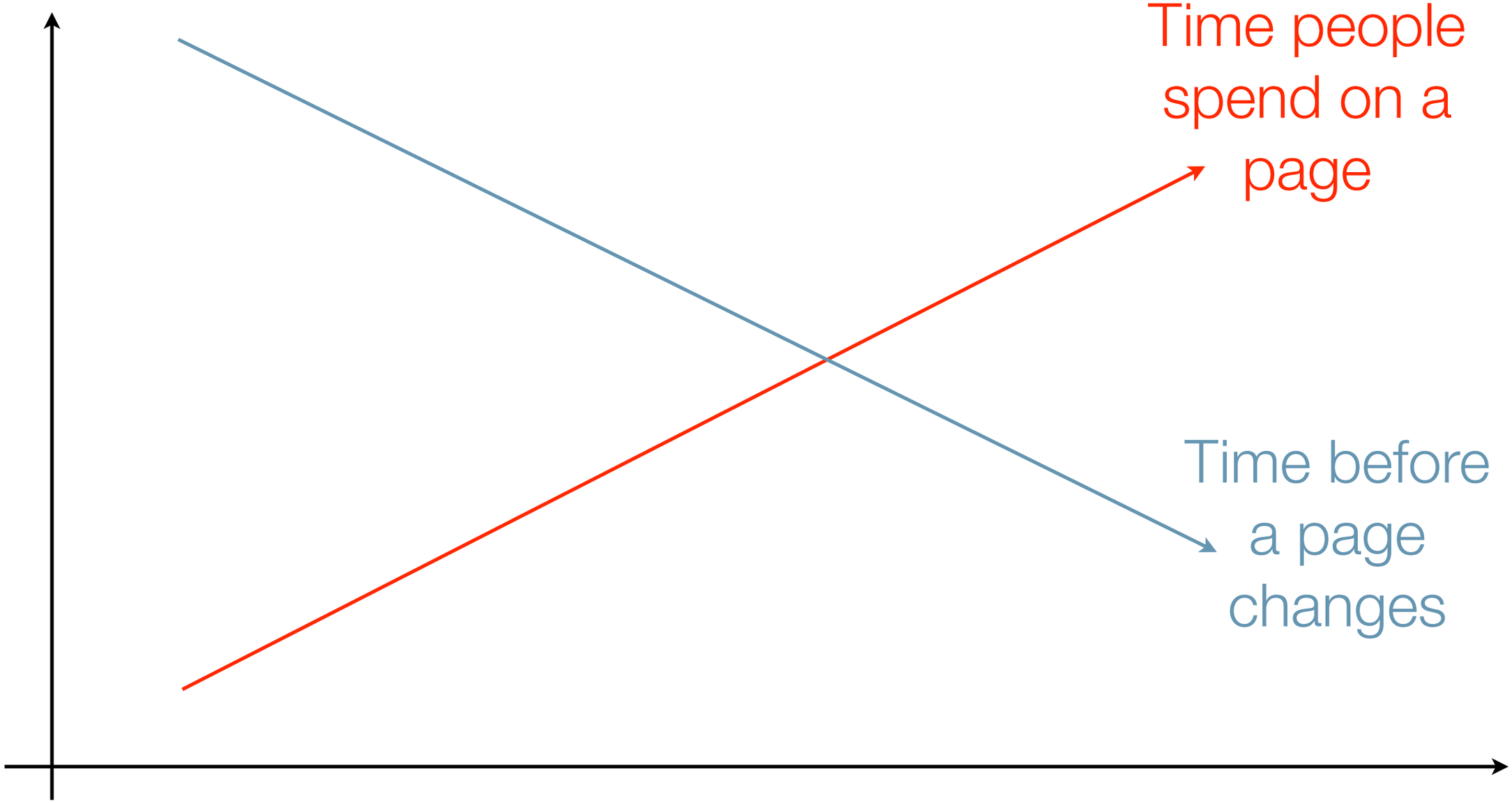
- Low latency data
- For events outside the browser

Why?

Ajax made individual pages interactive places to explore

More and more of the data on the web is social and therefore changing

Why?


Time people spend on a page
Time before a page changes
Evolution of the Web

Comet turns the web into a network of people who interact more naturally than they could behind a request/response barrier.

Examples of Comet

Chat is everywhere: GMail, Meebo, Yahoo Mail ...

Collaborative Editing: GDocs, Thinkature, Jot

Streaming Financial Data: Lightstreamer, Caplin

Asynch Updates: GMail, Yes.com

Online Gaming: GPokr

Async Server Processing: Polar Rose

Agenda:

Comet 101

The Ultimate Hack

Passing the Pain

DWR 101

Demos

sitepen

But ...

It's a hack - the web is biased against it

Client Issues

Maximum of 2 connections per browser per host (IE7/6)

- Coordination using `window.name` in the client
- or cookies using a server
- or use multi-home DNS

HTTP streaming is download only (chunked mode)

TCP connections are kept alive under HTTP 1.1

Server detection of failed connections

Client How-to: Forever Frame

Client posts an iframe which doesn't close quickly

- Send text/plain and poll in browser (not IE)
- Send text/plain with 4k whitespace to flush IE
- Flush with a <script> tag for each data block

The iframe will need killing and restarting to avoid memory leak

But IE clicks when iframe starts

Client How-to: Long Polling

Client makes an XHR request which does not return immediately

IE disallows reading XHR.responseText until connection is closed

Although you can keep XHR frames open forever, generally you poll

Client How-to: htmlfile

'htmlfile' is an ActiveX control like XHR:

```
htmlfile = new ActiveXObject("htmlfile");
htmlfile.open();
htmlfile.write("<html><iframe src='javascript:void(0)'
    onload='cleanup();'></iframe></html>");
htmlfile.close();
htmlfile.parentWindow.dwr = dwr;
```

Avoids 'clicking', but doesn't work in IE/Server 2003

Not supported in Firefox, Safari, Opera, etc.

Client How-to: Callback Polling

Create <script> blocks pointing to any domain

Create new script block when last completes

Client How-to: Other Options

Mime Messaging:

- Uses Multipart Mime in HTML: x-multipart-replace
- Not in IE
- Excellent performance

Server-Sent Events: WHATWG, but currently Opera only

Flash: We probably have enough other options that we don't need to get into plugins

Server Tricks

Watch out for stream-stoppers

- Apache: mod_jk
- Buggy network proxies
- Various application firewalls

Watch out for thread starvation

Does that stop us?

Ajax is also a hack, but that hasn't stopped it

And Comet does work

Comet vs. Polling

For Polling:

- More efficient for very low latencies
- Simpler to implement

For Comet:

- More efficient for all but very low latencies
- More adaptable

Agenda:

Comet 101

The Ultimate Hack

Passing the Pain

DWR 101

Demos

sitepen

Saving you the Pain

On the Server:

- Jetty, Twisted Python, Grizzly, Lighttpd, Perbal, Juggernaut
- Everyone except Apache

Event Buses

- Cometd, mod_pubsub, mod_repubsub, Lightstreamer, KnowHow, HAppS

Frameworks

- DWR, Juggernaut, Nevow

Bayeux

Standard Protocol for Interoperable Comet

Supported by:

- Cometd, Jetty, Dojo, DWR and servers in development from BEA, IBM and Sun

Bayeux on the Client via Cometd

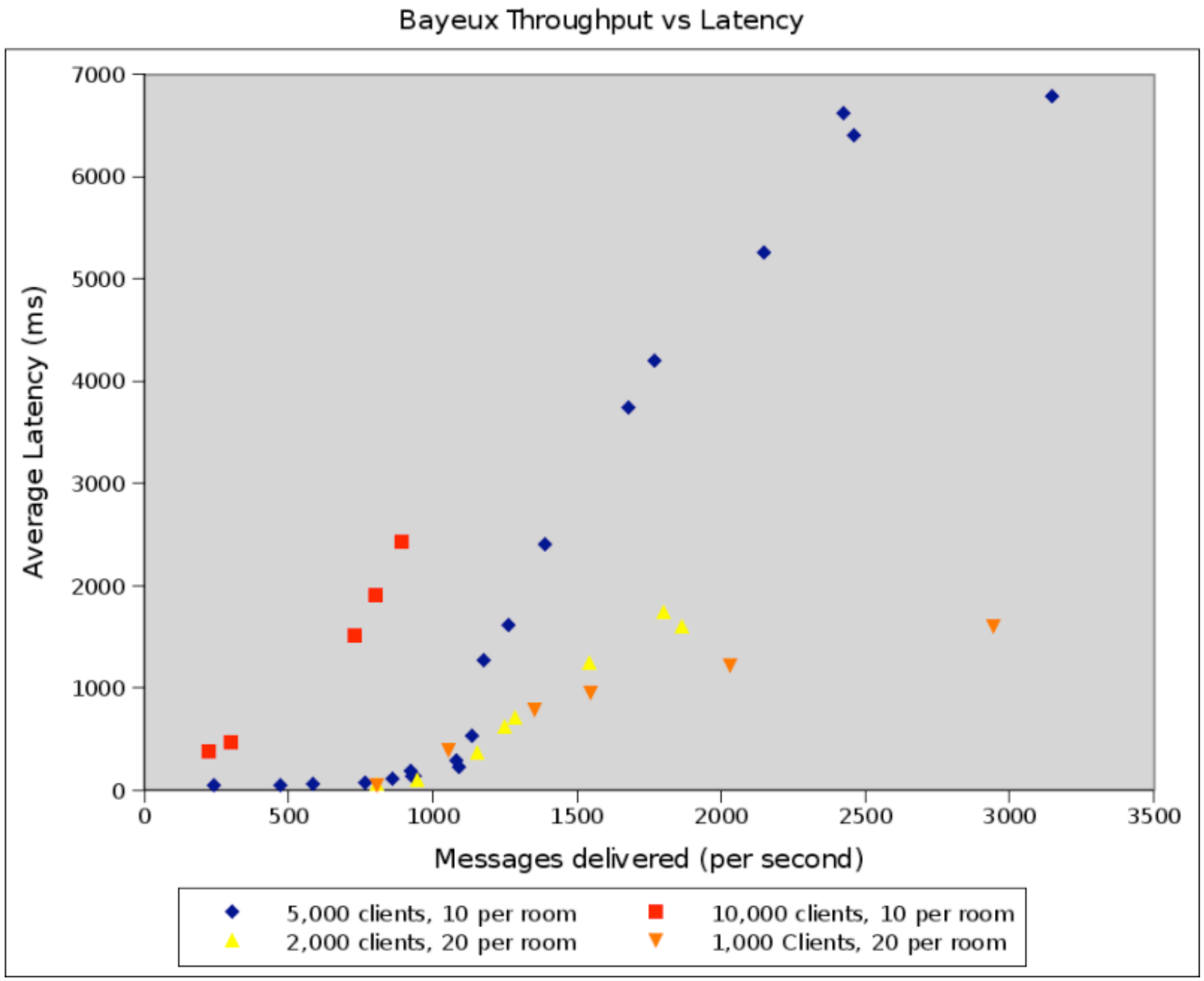
Dojo client implementation:

```
dojox.cometd.init(serverUrl);
dojox.cometd.publish("/topic", {/* payload */});
dojox.cometd.subscribe("/topic", function(){/* ... */ });
```

Bayeux on the Server

```
package dojox.cometd;

public interface Bayeux
{
  Client newClient(String idprefix, Listener listener);

  void publish(Client fromClient, String toChannel,
               Object data, String msgId);

  void subscribe(String toChannel, Client subscriber);

  ...
}
```

Bayeux Performance



Agenda:

Comet 101

The Ultimate Hack

Passing the Pain

DWR 101

Demos

sitepen

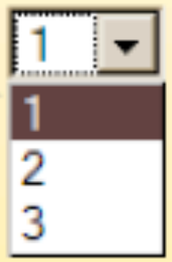
Web Browser

Web Server

HTML / Javascript

```
function eventHandler() {
    AjaxService.getOptions(populateList);
}
```

```
function populateList(data) {
    dwr.util.addOptions("listid", data);
}
```



Java

```
public class AjaxService {

  public String[] getOptions() {
    return new String[] { "1", "2", "3" };
  }

}
```

On the Server

```
public class SomeObject {

  public String sayHello(String n) {
    return "Hello, " + n;
  }

  public Set<Car> find(long reg) {
    // ...
    return cars;
  }
}
```

On the Server

```
@RemoteProxy
public class SomeObject {
  @RemoteMethod
  public String sayHello(String n) {
    return "Hello, " + n;
  }
  @RemoteMethod
  public Set<Car> find(long reg) {
    // ...
    return cars;
  }
}
```

On the Server

```
public class SomeObject {

  public String sayHello(String n) {
    return "Hello, " + n;
  }

  public Set<Car> find(long reg) {
    // ...
    return cars;
  }
}
```

On the Client

```
<script
  src='dwr/interface/SomeObject.js'>
```

On the Server

```
public class SomeObject {

  public String sayHello(String n) {
    return "Hello, " + n;
  }

  public Set<Car> find(long reg) {
    // ...
    return cars;
  }
}
```

On the Client

```
<script
  src='dwr/interface/SomeObject.js'>

<script>
SomeObject.sayHello('Joe', update);

function update(data) {
  $('field').innerHtml = data;
}
</script>
```

On the Server

```
public class SomeObject {

  public String sayHello(String n) {
    return "Hello, " + n;
  }

  public Set<Car> find(long reg) {
    // ...
    return cars;
  }
}
```

On the Client

```
<script
  src='dwr/interface/SomeObject.js'>

<script>
SomeObject.find('a1', function(cars) {
  dwr.util.addRows('table', cars, ...);
});


</script>
```

DWR can marshall:

Primitive types, and their Object counterparts
int, boolean, long, float, double, etc

Obvious classes
(String, Date, BigDecimal, BigInteger, Enum, etc)

Arrays and Collections (Map, List, Set, Iterator, ...)

JavaBeans and Objects

XML objects (DOM, XOM, JDom, Dom4J)

Images and Binary Files in version 3.0

(in short, basically anything ...)

Web Browsers

Web Server

page.html

page.html

page.html

1

1
2
3

1

1
2
3

DWR

Java

```
public class Monitor {

  public void asyncEvent(String value) {
    Collection sessions =
      wctx.getScriptSessionsByPage(...);
    Util util = new Util(sessions);
    util.setValue("listid", value);
  }

}
```

Terms

Comet = long lived HTTP connections

Polling = regular short HTTP connections

Piggyback = data smuggled on other connections

Reverse Ajax = any of the above

On the Client

share.html

```
<p>Share price:
  <span id='price'>
  </span>
</p>
```

On the Client

share.html

```
<p>Share price:
  <span id='price'>
  </span>
</p>
```

On the Server

```
ServerContext ctx = ServerContextFactory.get();
Collection sessions =
  ctx.getScriptSessionsByPage("share.html");
```

On the Client

share.html

```
<p>Share price:
  <span id='price'>
  </span>
</p>
```

On the Server

```
ServerContext ctx = ServerContextFactory.get();
Collection sessions =
  ctx.getScriptSessionsByPage("share.html");

String s = "dwr.util.setValue('price', 42)";
ScriptBuffer b = new ScriptBuffer(s);

for (ScriptSession session : sessions) {
  session.addScript(b);
}
```

On the Client

share.html

```
<p>Share price:
  <span id='price'>
  </span>
</p>
```

On the Server

```
ServerContext ctx = ServerContextFactory.get();
Collection sessions =
  ctx.getScriptSessionsByPage("share.html");


Util pages = new Util(sessions);
pages.setValue("price", 42);
```

On the Client

share.html

```
<p>Share price:
  <span id='price'>
  </span>
</p>
```

On the Server

```
ServerContext ctx = ServerContextFactory.get();
Collection sessions =
  ctx.getScriptSessionsByPage("share.html");

import org....scriptaculous.Effect

Effect e = new Effect(sessions);
e.shake("price");
```

On the Client

share.html

```
<p>Share price:
  <span id='price'>
  </span>
</p>
```

On the Server

```
ServerContext ctx = ServerContextFactory.get();
Collection sessions =
  ctx.getScriptSessionsByPage("share.html");

import jsx3.gui.*

Server s = GI.getServer(sessions, "app");
Button b = s.getJSXById("id", Button.class);
b.setEnabled(Form.STATEDISABLED, true);
```

On the Client

share.html

```
<p>Share price:
  <span id='price'>
  </span>
</p>
```

On the Server

```
ServerContext ctx = ServerContextFactory.get();
Collection sessions =
  ctx.getScriptSessionsByPage("share.html");

import org.dojotoolkit.dijit.Dijit;
import org.dojotoolkit.dijit.Editor;

Dijit dijit = new Dijit(sessions);
Editor e = dijit.byId("price", Editor.class);
e.setValue(42);
```

New in DWR 3.0

Converters: Image/Binary file upload/download

Busses: JMS, OpenAjax Hub

JSON / Bayeux support

Drapgen: GI and maybe Dojo in Java

Gears: Auto offline

Pub-Sub

```
Hub hub = HubFactory.get();

hub.publish(TOPIC_TEXT, info);
```

```
hub.subscribe(TOPIC_PEOPLE, new MessageListener() {
    public void onMessage(MessageEvent message) {
        Person p = message.getData(Person.class);
        // ...
    }
});
```

Gears

For a certain set of applications, offline comes free

Remote calls can be optional

- ‘Refresh’ is a common optional operation
- ‘Save’ is a common mandatory option

If offline, DWR will queue the mandatory ones for later, and let you carry on without the optional ones.

Agenda:

Comet 101

The Ultimate Hack

Passing the Pain

DWR 101

Demos

sitepen

Questions?

http://cometdaily.com/
http://cometd.com/
http://getahead.org/blog/joe/
http://directwebremoting.org/

sitepen