Real-World Comet-Based Applications

Jean-Francois Arcand, Sun Microsystems, Inc.

Greg Wilkins, Webtide

Alex Russel, DOJO Foundation

TS-6807

Goal of This Talk

What You Will Gain

This session will introduce Comet technologies. Comet (sometimes called request polling, HTTP streaming, or continuation) is a programming technique that enables web servers to send data to the client without any need for the client to request it. It allows creation of event-driven web applications.

Agenda

- Introduction
- Introduction to Comet
- The need for standards
    - Bayeux
    - The Open Ajax Alliance
- Available solutions
- Q&A

Agenda

Introduction

Introduction to Comet

The need for standards

- Bayeux
- The Open Ajax Alliance

Available solutions

Q&A

Introduction

Who we are....

- Jean-Francois Arcand, Senior Staff Engineer
  - Main developer of the Project Grizzly NIO framework and Project GlassFish™ WebContainer
- Greg Wilkins, CTO, Webtide
  - Main developer of the Jetty HTTP server and servlet container
- Alex Russell—Director of R&D at SitePen

Agenda

- Introduction
- **Introduction to Comet**
- The need for standards
    - Bayeux
    - The Open Ajax Alliance
- Available solutions
- Q&A

Introduction to Comet

- Comet is:
  - Programming technique for long-running connections over HTTP
  - Allows creation of event-driven, interactive web apps

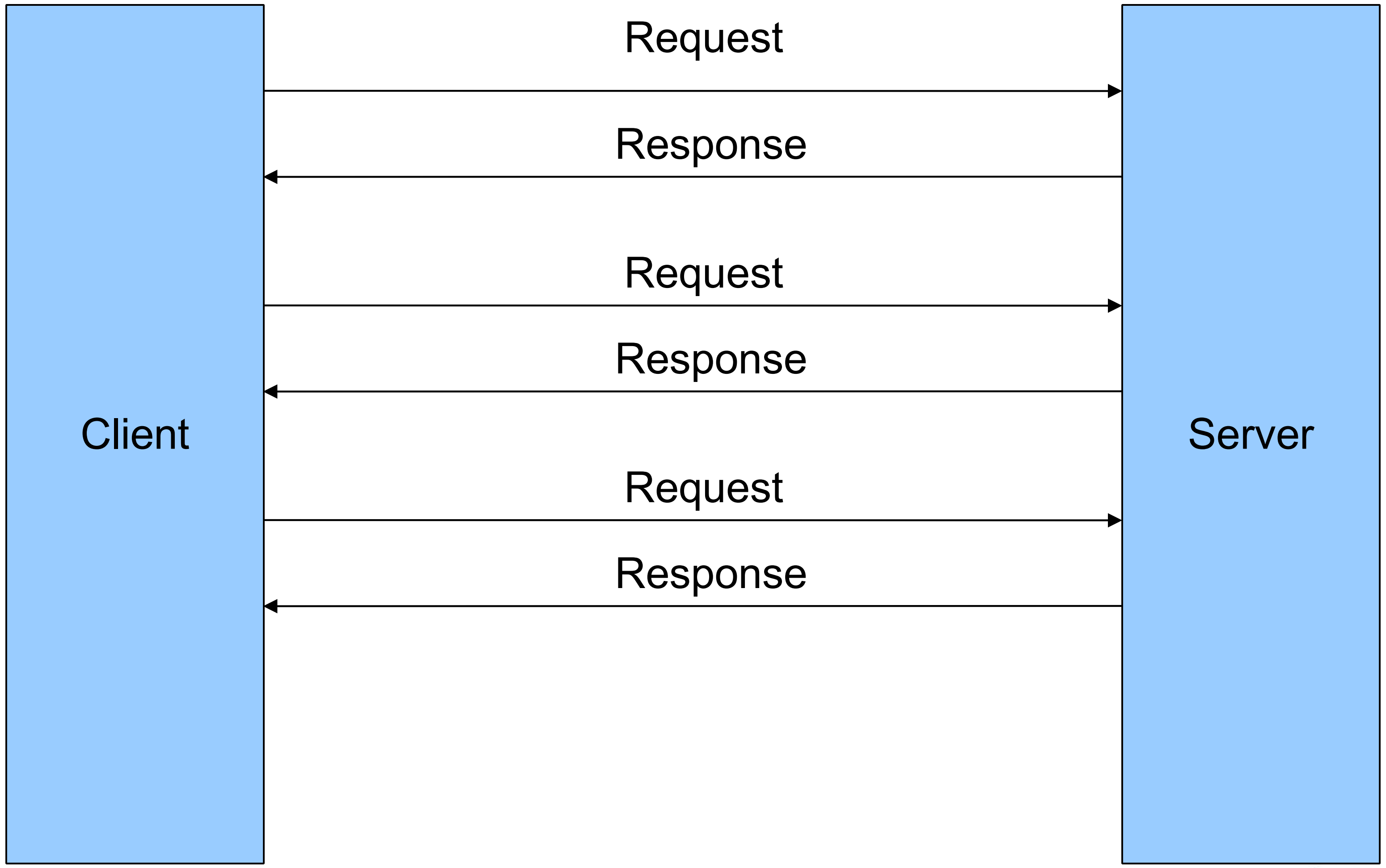
What Makes Comet Application Different From Traditional Application?

- Fundamentally, they all use long-lived HTTP connections to reduce the latency with which messages are passed to the server
- In essence, they do not poll the server occasionally; Instead the server has an open line of communication with which it can push data to the client

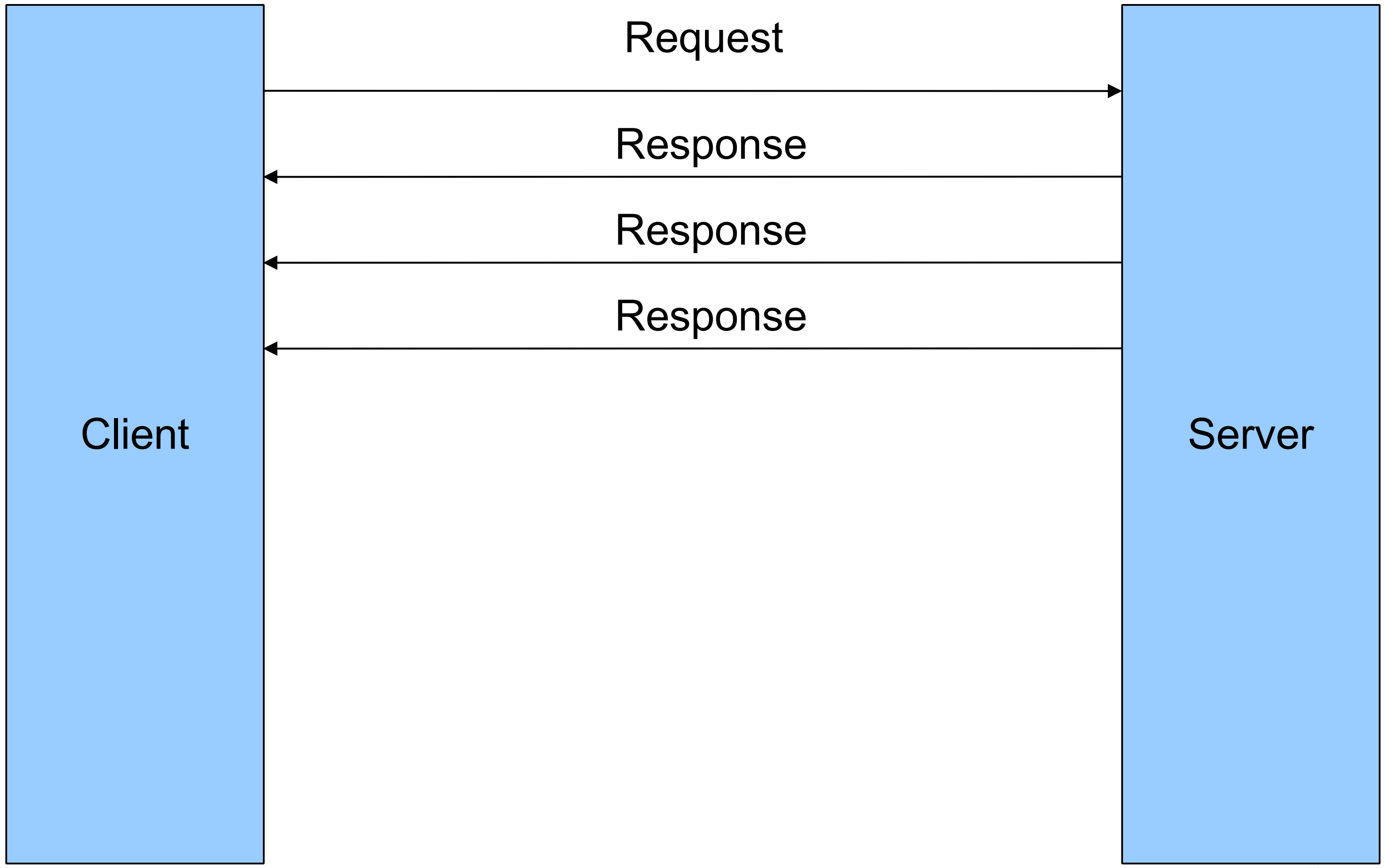
What Makes Comet Application Different From Traditional Application?

- Comet applications can deliver data to the client at any time, not only in response to user input—The data is delivered over a single, previously-opened connection
- This approach reduces the latency for data delivery significantly

Before Comet: Polling for Content



With Comet: Pushing Content



Agenda

Introduction

Introduction to Comet

The need for standards

- Bayeux
- The Open Ajax Alliance

Available solutions

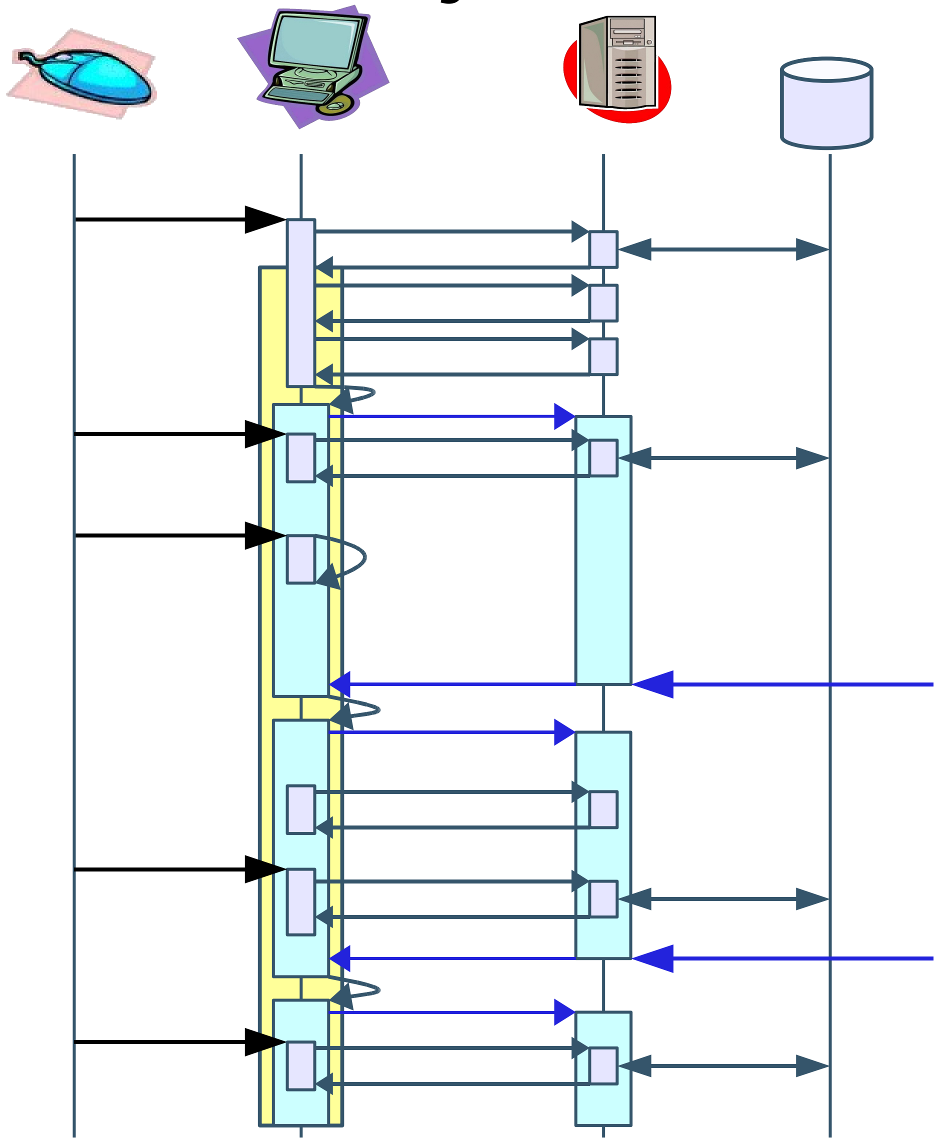
Q&A

The Need for Comet Standards

Standard == interoperability

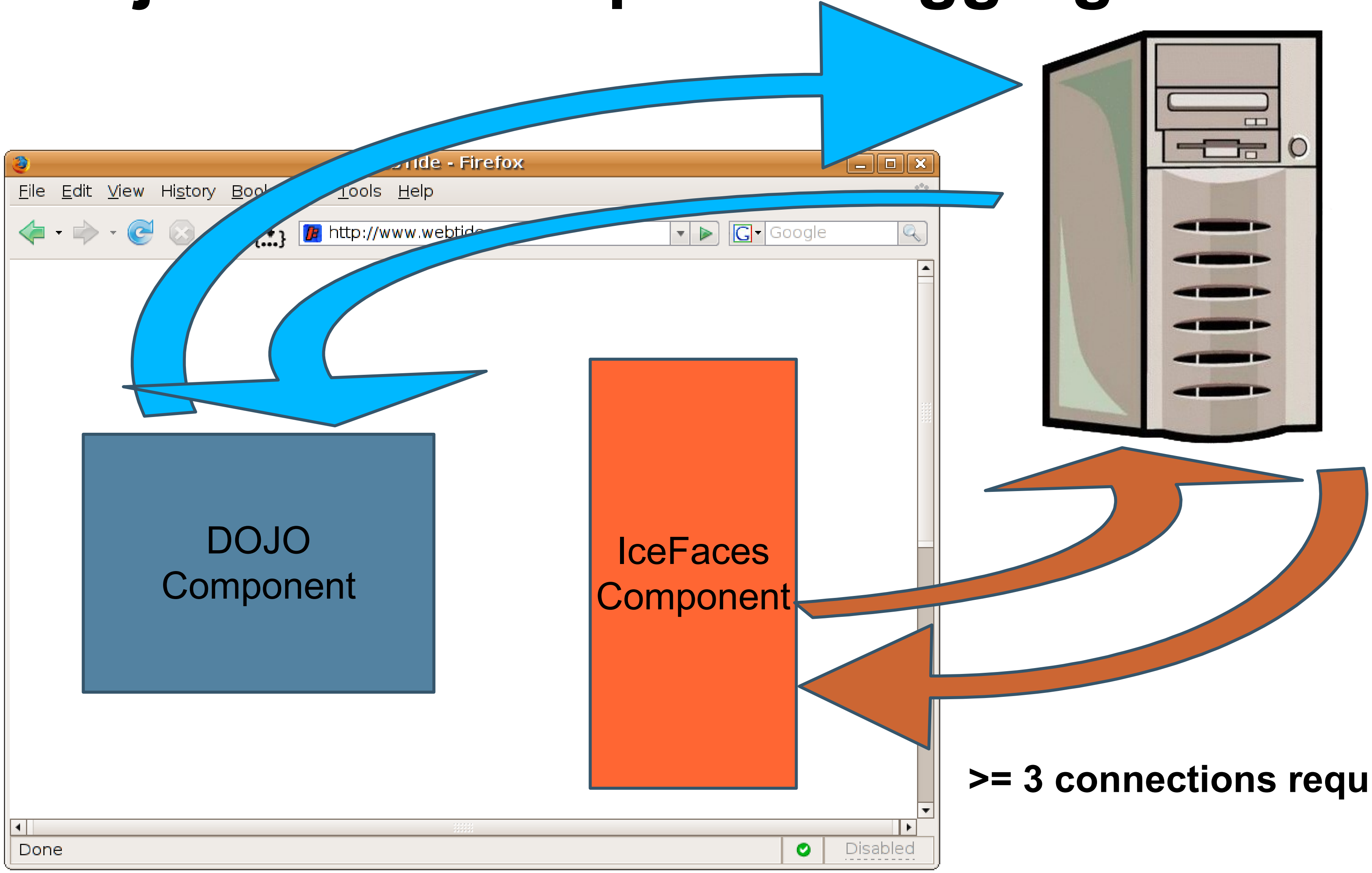
- Ajax components work on limited resources
  - JavaScript™ technology namespace
  - IU
  - 2 connections per host (Internet Explorer restriction)
- Requirement to mix Ajax frameworks
  - No one-size-fits-all framework
  - Portlets and mashups
  - Multi-tabs and windows

Comet Ajax Push—Interaction



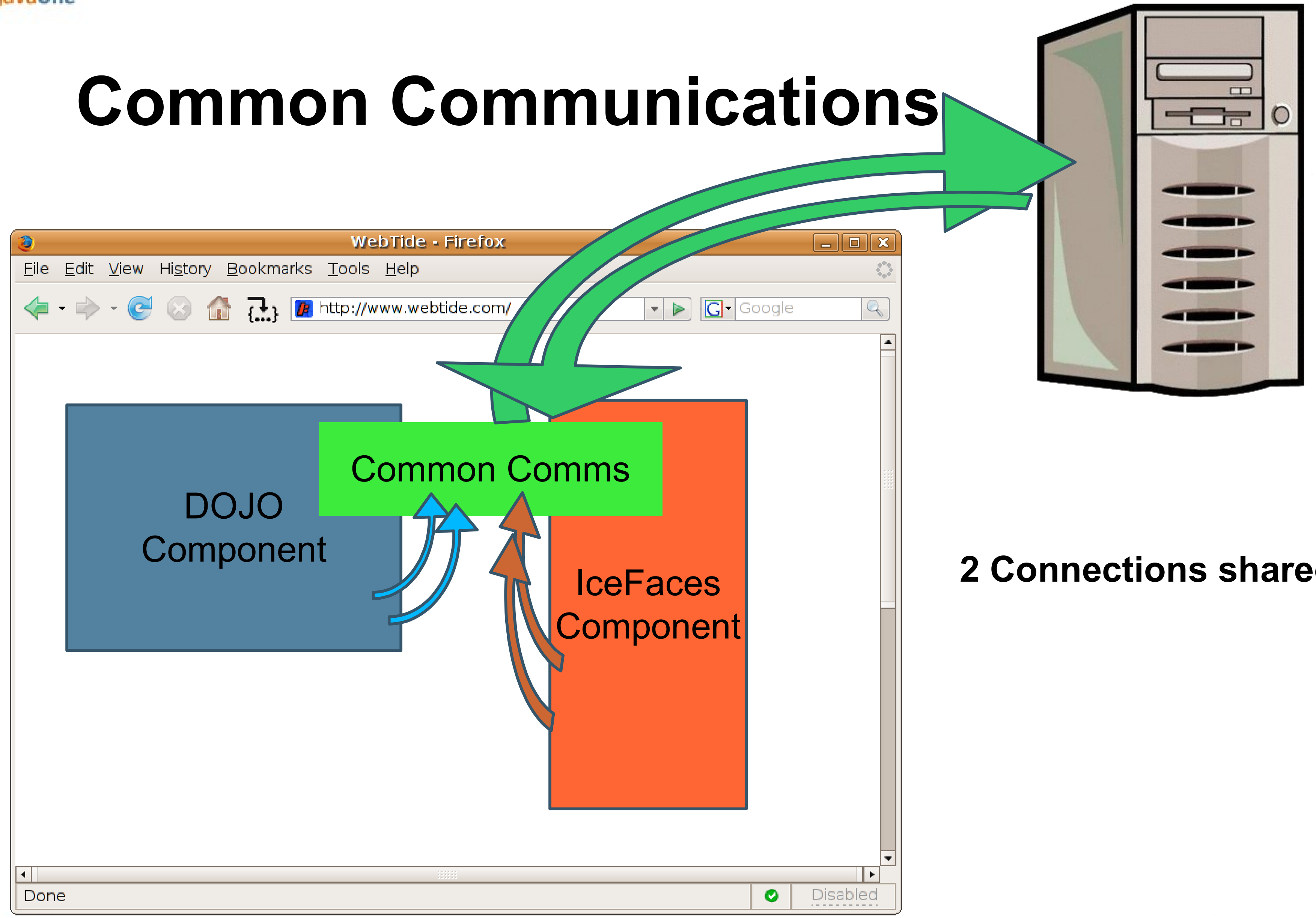
- Bidirectional
- Asynchronous
- Outstanding request waits for server events
- Two connections in use
  - Per browser per host

Ajax Push Component Aggregation



>= 3 connections required!

Common Communications



2 Connections shared

How to Get Everybody to Agree?

- Would everybody use the same framework?
  - Maybe a framework framework!
- JSON vs. XML wars
- Long poll vs. forever frame vs. iframes vs. XHR vs. JS callback vs...
- What about cross domain?
- Shouldn’t we wait for the browsers to help?
- Will one size every fit all?
- Security?

Working Both Ends of the Problem!

- Frames works
  - Implementing Comet protocols:
    - e.g., DOJO Foundation—Bayeux
      - An implementation of a protocol
- Open Ajax Alliance—Open Ajax Hub
  - A specification for interoperability
    - With a reference implementation

Agenda

Introduction

Introduction to Comet

The need for standards

- **Bayeux**
- The Open Ajax Alliance

Available solutions

Q&A

What Is the Bayeux Protocol?

- Bayeux is a negotiated, multi-transport, JSON-based protocol for publish/subscribe asynchronous messaging between:
    - Web browser and server
    - Server and web browser
    - Web browser and web browser
- The goals of the Bayeux spec
    - Make event delivery fast
    - Keep it simple
    - Provide extension points in the protocol

The Bayeux Protocol

- Part of the Dojo Cometd project
  - HTTP-based event routing bus that uses a push technology pattern known as Comet
  - http://www.cometd.com
- Negotiated Multi-transport JSON-based Publish/Subscribe Messaging
- Multiple implementations
  - JavaScript programming language, Java™ programming language, perl, python

What Is Cometd?

- Cometd is implementation of the Bayeux protocol
- Cometd consists of a protocol spec called Bayeux, JavaScript technology libraries (DOJO toolkit), and an event router
- The Server uses async techniques to “park” the request without blocked threads
- Comet programming is easy, cometd is even easier
  - No server-side elements need to be created

Bayeux: Publish/Subscribe API

Bayeux: JSON Based

```
[
 {
    "channel": "/meta/connect",
    "successful": true,
    "error": "",
    "timestamp": "12:00:00 1970"
  },
  {
    "channel": "/some/channel",
    "data": "some data"
  }
]
```

Bayeux: JSON vs. XML???

- Had to pick one
  - Better fit for JavaScript technology clients
- Security issues address
  - Commented JSON format supported
  - Cookies/sessions not used for authentication
  - Auth tokens exchanged in packets
- Alternative transports supported
  - JSON used only to define message content
  - A transport may choose to encode JSON objects as XML on the wire!

Bayeux: Multi-Transport

- Standard transports
  - Long-polling: default
  - Callback-polling: cross domain
- Alternative transports
  - iFrame
  - Mime-message-block
  - Polling
- Encourage innovation with transports
  - Future standards?
  - Browser support?

Bayeux: Long-Polling vs. Forever Response

- Long-polling conforms to HTTP RFC
  - Forever response assumes no caching proxies
- HTTP/1.1 persistent connections
  - No tear down and reconnect at TCP/IP level
- Long-polling data framed with HTTP
  - Forever response data needs framing (mime or JavaScript technology) and/or flushing data
- Poll requests can piggyback on ACKs
  - Forever response data chunks need to be ACKed

Bayeux: Security

- No JavaScript technology hijacking
  - No autobrowser authentication
  - Commented JSON
  - In band tokens
- Reduced attack surface
  - All (most) communications is routed via bayeux channels which may have per channel filters for:
    - No markup
    - No scripts
    - SQL injection detection
    - Data validation

DEMO

Bayeux Demo Using Jetty and Project GlassFish

Agenda

Introduction

Introduction to Comet

The need for standards

- Bayeux
- **The Open Ajax Alliance**

Available solutions

Q&A

The Open Ajax Alliance

- Promoting ability to match solutions from Ajax technology providers

  24SevenOffice, ActiveGrid, ActiveState, Adobe, American Greetings, Appeon, Aptana, Arimaan Global Consulting, BEA, Curl, Custom Credit Systems (Thinwire), Dojo Foundation, Eclipse Foundation, edge IPK, eLink Business Innovations, ESRI, Finetooth, Getahead (DWR), Global Computer Enterprises, GoETC, Google, Helmi Technologies, HR-XML, IBM, ICEsoft, Ikivo, ILOG, Innoopract, iPolipo, Isomorphic Software, IT MILL, JackBe, Javeline, JSSL, JWAX, Laszlo, Systems, Lightstreamer, Microsoft, MobileAware, Mozilla Corporation, NetScript Technologies, Nexaweb, Nitobi, Novell, OpenLink Software, OpenSpot, OpenSymphony (OpenQA), Openwave Systems, Opera, OpSource, Oracle, OS3.IT, Redmonk, SAP,Scalix, Seagull Software, Sitepen, Software AG, Sun Microsystems, Tealeaf Technology, Teleca Mobile, The Frontside, TibcoTransmend, Vertex Logic, Visible Measures, Visual WebGui, Volantis Systems, Webtide, XML11, Zend, Zimbra, Zoho
- http://www.openajax.org

Open Ajax Hub—Communications

- Captures Semantics, not implementations
  - Request/Response
  - RPC
  - Publish/Subscribe

Open Ajax Hub—Communications

- Connection-based API
  - Connection is an abstraction of transport and configuration
  - Connections accessed by ID and/or URL
  - Connections are:
    - Configured by Page Composer
    - Used by Component developers
    - Implemented by Communications providers

Open Ajax Hub—Simple API

- Request Response
  - connection.request(destination,payload,callback);
- Publish Subscribe
  - connection.subscribe(destination,payload,callback);
  - connection.unsubscribe(destination);
  - connection.publish(destination,payload);
- Batching
  - connection.startBatch();
  - connection.endBatch();

XHR Provider

- Maps API to XmlHttpRequest
- Destination == URI
- Pluggable payload encode/decode
  - Allows mapping to existing web servers
- Subscribe is implemented with poll
  - May be long if server implemented

Bayeux Provider

- Maps API to cometd bayeux
- Destination == Channel
- Data sent unencoded
- Server-side clients may convert to existing web servers
- Multi-transport for polling

DEMO

Open Ajax

Agenda

Introduction

Introduction to Comet

The need for standards

- Bayeux
- The Open Ajax Alliance

Available solutions

Q&A

Solution: Project Grizzly Bayeux

- Based on Project Grizzly’s Asynchronous Request Processing extension
  - The Cometd/Bayeux protocol is implemented as a Project Grizzly’s Comet application, called gCometd
  - Services (database, Web Service, Web Application) can extends the cometd framework and push data back to Ajax clients without the needs to understand the Bayeux protocol
- Available in GlassFish V2 and Grizzly HTTP server

DEMO

Project jMaki on Project Grizzly

Solution: Jetty Bayeux

- Based on Jetty Continuation mechanism
    - Allows the current threads handler to be suspended and resumed at a later time in response to a time out or an asynchronous event
    - A continuation is obtained by a servlet via a portable API that will work in any Servlet container like Apache Tomcat, Resin, Project GlassFish, and of course, Jetty
- Was the first Java Web Server™ to support both Comet and Cometd/Bayeux
    - Strong community using it

Summary

- Does your application frequently poll your server? Try Comet to reduce the latency and reduce the load on your server
- Want to use a standard protocol? Learn Bayeux!
- Growing number of libraries already support or will soon support Comet/Cometd natively (e.g., DOJO Toolkit, Project jMaki, etc.)

For More Information

- Jetty: http://www.webtide.com/
- Project Grizzly: http://grizzly.dev.java.net
- Project GlassFish: http://glassfish.dev.java.net
- DOJO Foundation: http://dojotoolkit.org/
- OpenAjax: http://www.openajax.org/

Q&A

Real-World Comet-Based Applications

Jean-Francois Arcand, Sun Microsystems, Inc.

Greg Wilkins, Webtide

Alex Russel, DOJO Foundation

TS-6807