
TM
KAAZING

KAAZING

Breaking Barriers with HTML 5 Communication

How to enable a stateful Web

Challenge

"If we were not restricted by the traditional limitations of HTTP, what type of Web applications would we build?"

Speakers

- Jonas Jacobi
- Co-Founder: Kaazing
- Co-Author: Pro JSF and Ajax, Apress

Networking Review

- Desktop Networking
  - Full-duplex bidirectional TCP sockets
  - Access any server on the network
- Browser Networking
  - Half-duplex HTTP request-response
  - HTTP polling, long polling, streaming
  - Same-origin HTTP requests

Defining Real-Time Web

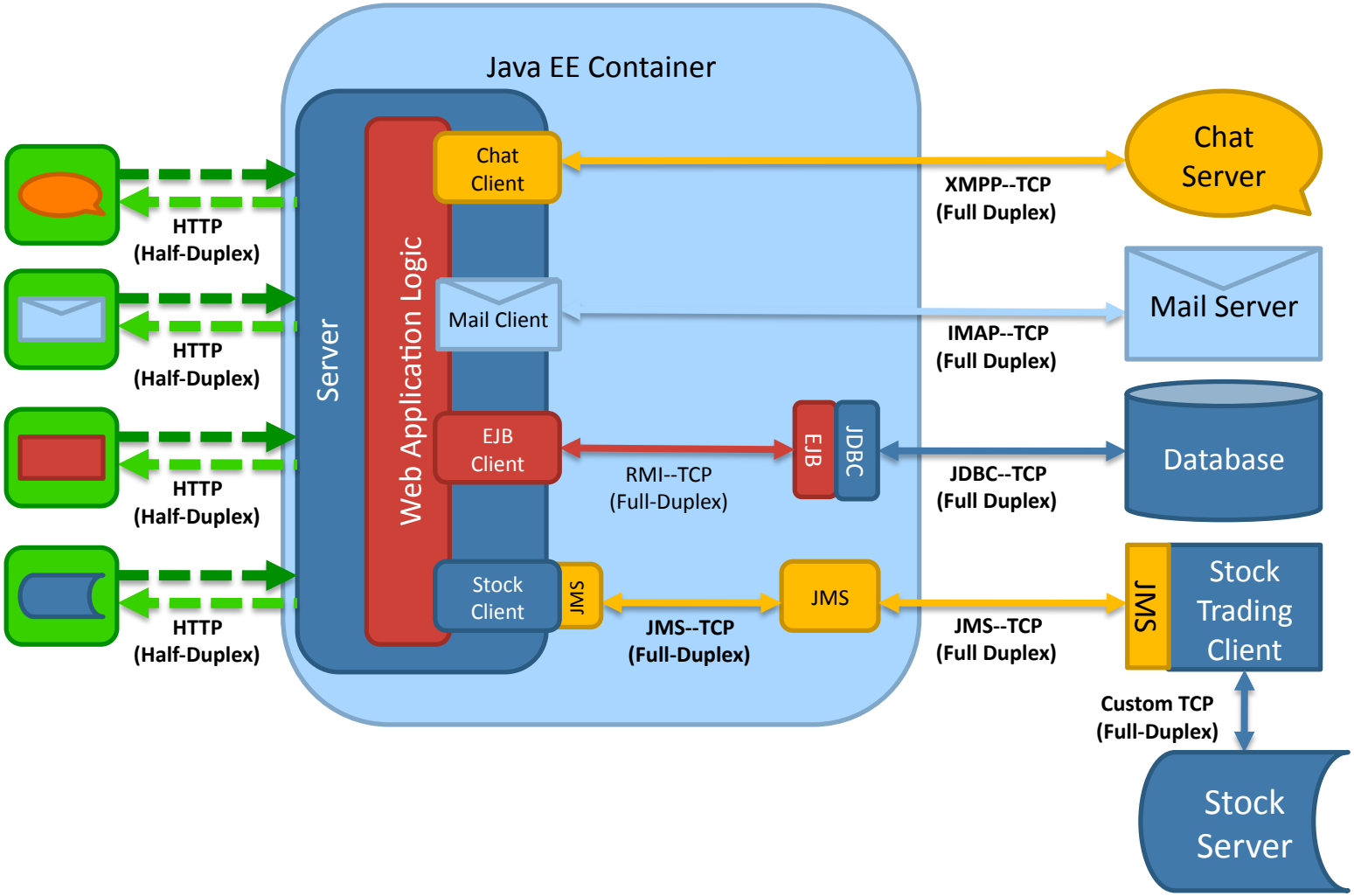
Or, is it just nearly, nearly real-time?

Push Technology

- Server-Initiated Message Delivery
  - Clients are listening
  - Clients behind firewalls
- Techniques such as Comet/Reverse Ajax
- Delays Completion of HTTP Response
- Generally Implemented in JS
- Scalability Limitations (Cost etc…)
- Not general purpose
- No standard

KAAZING

Half-duplex Architecture



W3C & IETF Work in Progress

- W3C
  - HTML5 Specification (postMessage)
  - Cross-Origin Resource Sharing
  - EventSource API (Server-sent Events)
  - WebSocket API
- IETF
  - WebSocket Protocol
    (in conjunction with W3C WebSocket API spec)

HTML 5 postMessage

- Send Strings Between HTML Documents
  - Documents may be served by different sites
- Standard API

```
targetWindow.postMessage(message,
                         targetOrigin)

window.onmessage = function(event) {
     alert(event.data);
  }
```

- Browser Support
  - IE 8, FF 3, Opera 9, Safari 4, Chrome 2

HTML 5 Server-Sent Events

- Standardizes and formalizes how a continuous stream of data can be sent from a server to a browser
- Introduces EventSource—a new JavaScript API

HTML 5 Server-Sent Events

Connects to a server URL to receive an event stream:

```
var stream =
  new EventSource("http://news.kaazing.com");

stream.onopen = function() { alert(“open”); }
stream.onmessage = function(event) {
  alert(“message: “ + event.data); }
stream.onerror = function() { alert(“error”); }
```

HTML 5 Server-Sent Events

- Server can add the id event property so that clients can add a Last-Event-ID header during reconnect
- Used to guarantee message delivery
- Server can specify an optional retry header as part of an event in the event stream

KAAZING

Cross-Site Resource Sharing

- W3C Technical Report
  - Access control for client-side cross-origin requests
  - Published Sept 12, 2008
  - http://www.w3.org/TR/access-control/
- Browser Support
  - Firefox 3.5
  - IE8 XDomainRequest (similar)
  - Opera, Safari, Chrome coming

Cross-Site Resource Sharing

```
GET / HTTP/1.1\r\n
Host: www.w3.org\r\n
Origin: http://www.kaazing.com\r\n
…
\r\n

200 OK HTTP/1.1\r\n
Allow-Origin: http://www.kaazing.com\r\n
…
\r\n
```

DEMO

“postMessage, EventSource”

WebSockets

- W3C Specification – WebSocket API
  - http://dev.w3.org/html5/websockets/
- HTTP-friendly TCP for the browser
- Full-duplex bidirectional communication
  - Operates over a single socket
- Browser Support (coming)
  - Webkit – Bug 28844
  - Firefox – Bug 472529

WebSockets

- Distributed client-server architecture
  - No browser plug-ins
- Traverses proxies and firewalls seamlessly
  - HTTP CONNECT
- Allows authorized cross-origin communication
- Share port with existing HTTP content at different path

WebSockets

- Connection established by upgrading from the HTTP protocol to the WebSocket protocol
- WebSocket data frames can be sent back and forth between the client and the server in full-duplex mode

WebSockets

- Supports a diverse set of clients
- Cannot deliver raw binary data to JavaScript
  - Binary data is ignored if the client is JavaScript

WebSocket Schemes

```
ws://www.websocket.org/text

wss://www.websocket.org/encrypted-text
```

KAAZING

WebSocket Handshake

```
GET /text HTTP/1.1\r\n
Upgrade: WebSocket\r\n
Connection: Upgrade\r\n
Host: www.websocket.org\r\n
…\r\n

HTTP/1.1 101 WebSocket Protocol Handshake\r\n
Upgrade: WebSocket\r\n
Connection: Upgrade\r\n
…\r\n
```

WebSocket Frames

- Frames can be sent full-duplex
  - Either direction at any time
- Text Frames use terminator
  - `\x80Hello, WebSocket\0xff`
- Binary Frames use length prefix
  - `\x00\0x10Hello, WebSocket`
- Text and binary frames on same WebSocket

WebSockets API

- Creating a WebSocket instance:

```
var myWebSocket = new WebSocket
("ws://www.websocket.org");
```

WebSockets API

- Associating listeners:

```
myWebSocket.onopen = function(evt)
  { alert("Connection open ..."); };
myWebSocket.onmessage = function(evt)
  { alert( "Received Message:  "  +
  evt.data); };
myWebSocket.onclose = function(evt)
  { alert("Connection closed."); };
```

WebSockets API

- Sending messages:

```
myWebSocket.postMessage("Hello
  WebSocket!");
myWebSocket.disconnect();
```

Extending WebSockets

- Any TCP-based Protocol Works on WebSocket
  - JMS, AMQP, STOMP, XMPP, IMAP, AMQP, IRC, …
  - Custom Protocols
- Binary Protocols
  - Encode Binary as Text

KAAZING

WebSocket Security

- HTTP Security
  - 401 Not Authorized
  - Cross-domain communication
- Secure WebSockets
  - **wss**://www.websocket.org
  - SSL just like HTTPS
- Single Sign-on
  - HTTP credentials vs. protocol credentials
- Protocol Attacks
  - Validate protocol syntax and semantics

KAAZING

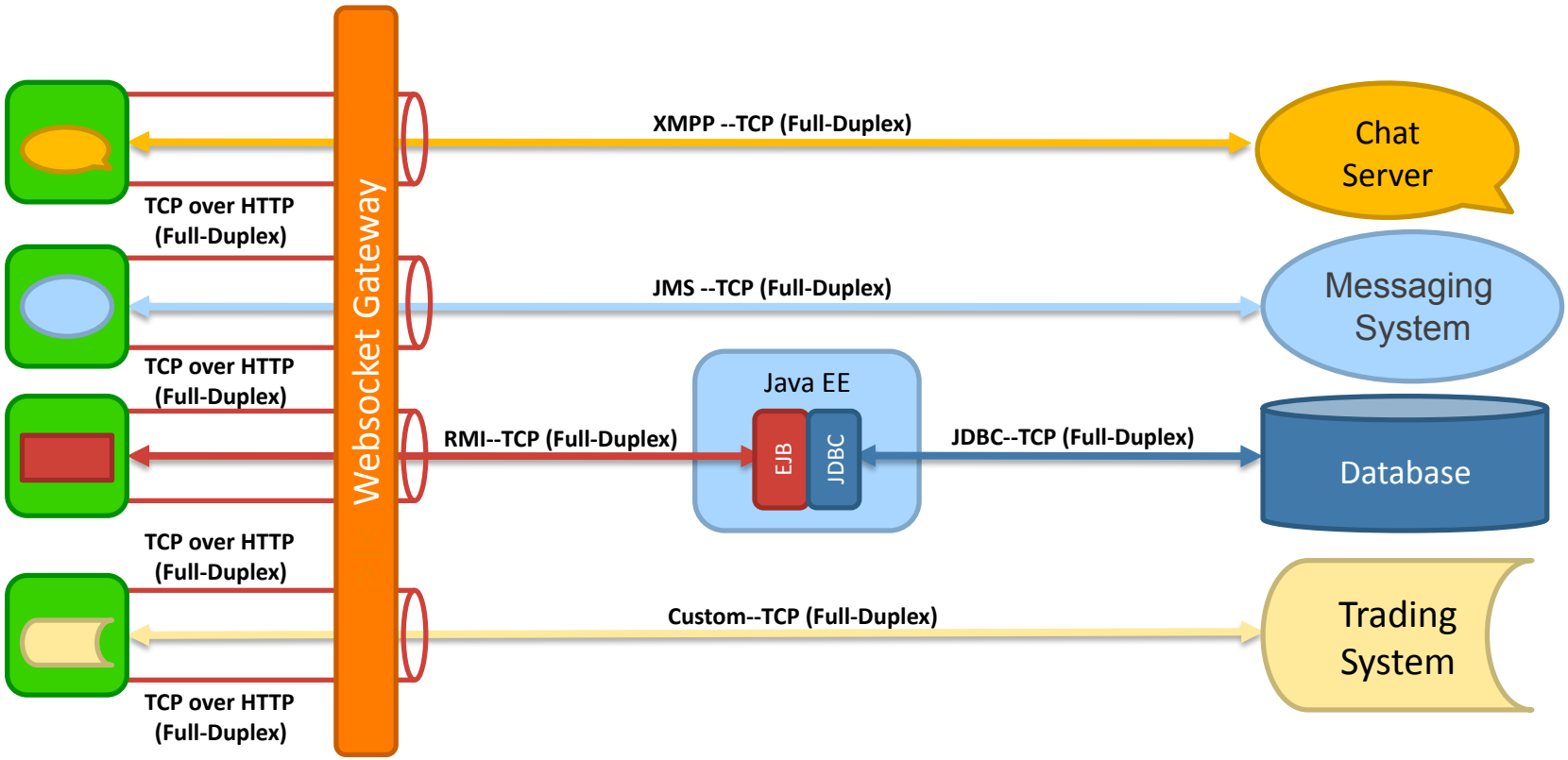
Stomp Client Example

```
var myStomp = new StompClient();

myStomp.onopen =
   function(headers) {
     myStomp.subscribe("/topic/destination");
   }
myStomp.onmessage =
   function(headers, body) { alert(body); }

myStomp.connect("ws://www.websocket.org/stomp");
myStomp.send("Hello STOMP!",
             "/topic/destination");
```

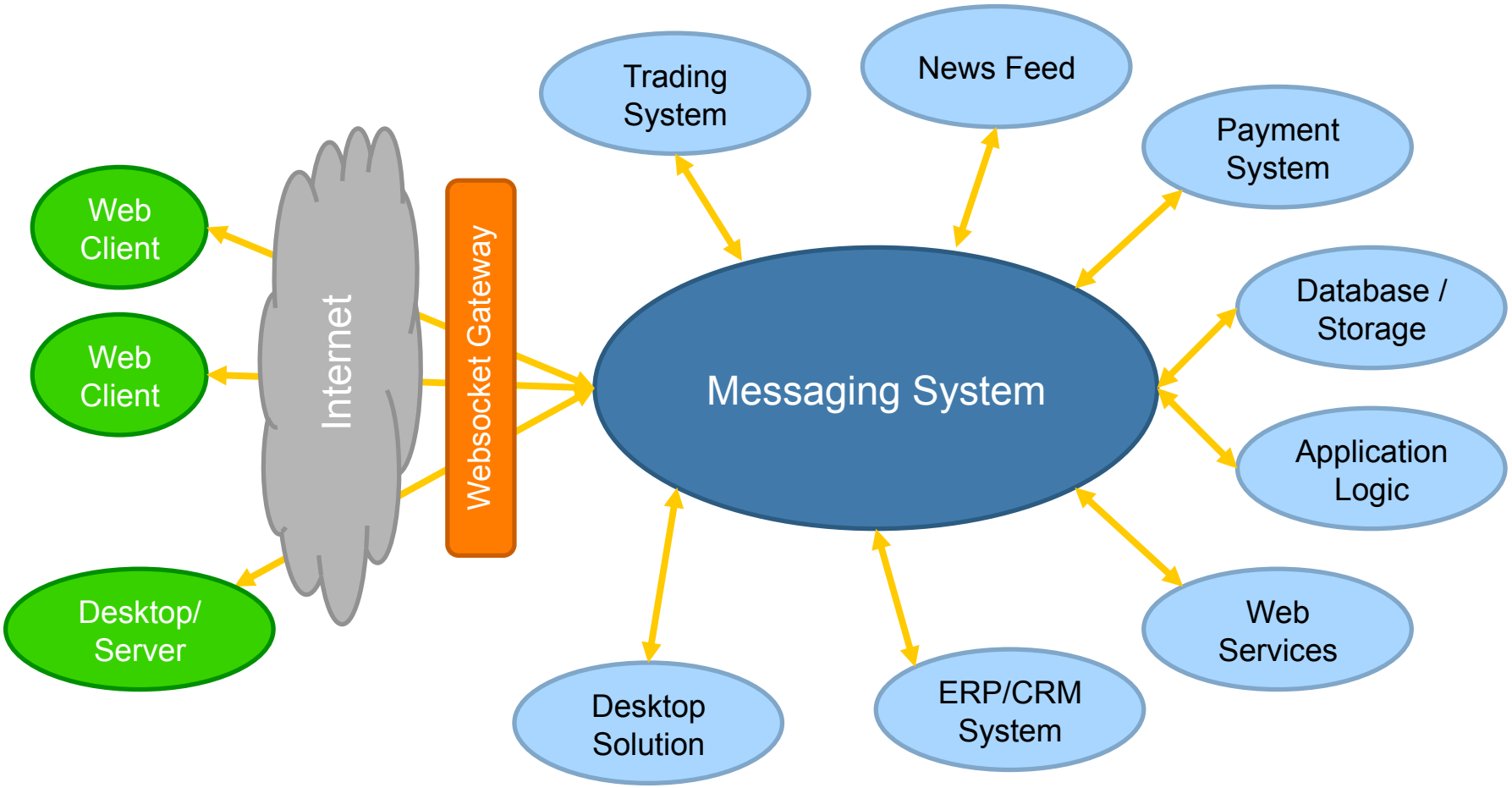
Full-duplex Architecture



DEMO

“WebSocket, Stomp, …”

Stateful Asynchronous System



Summary

- HTML 5 Communication has arrived (early)
- W3C WebSockets will change everything
- Open your mind and be creative

Q&A

WebSockets changes everything!

KAAZING

Coming soon to a bookstore …