AJAX & Geolocation

INF5750/9750 - Lecture 6 (Part IV)

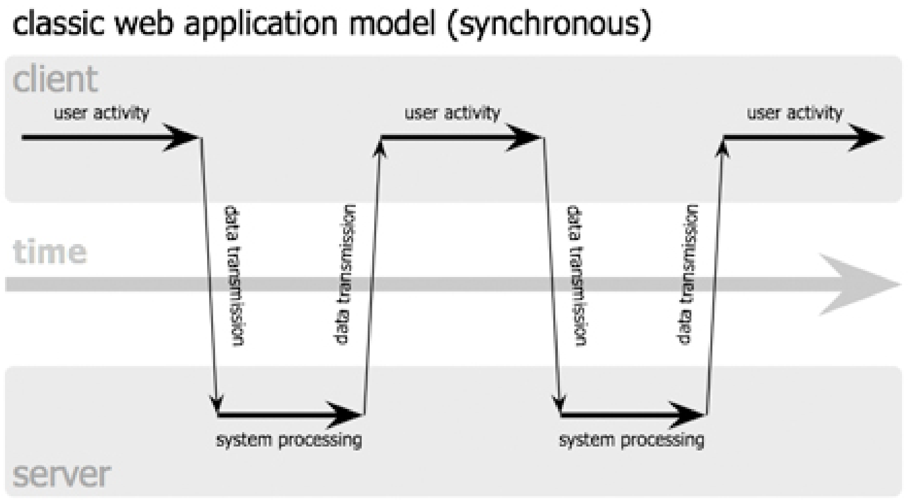
Problem Area

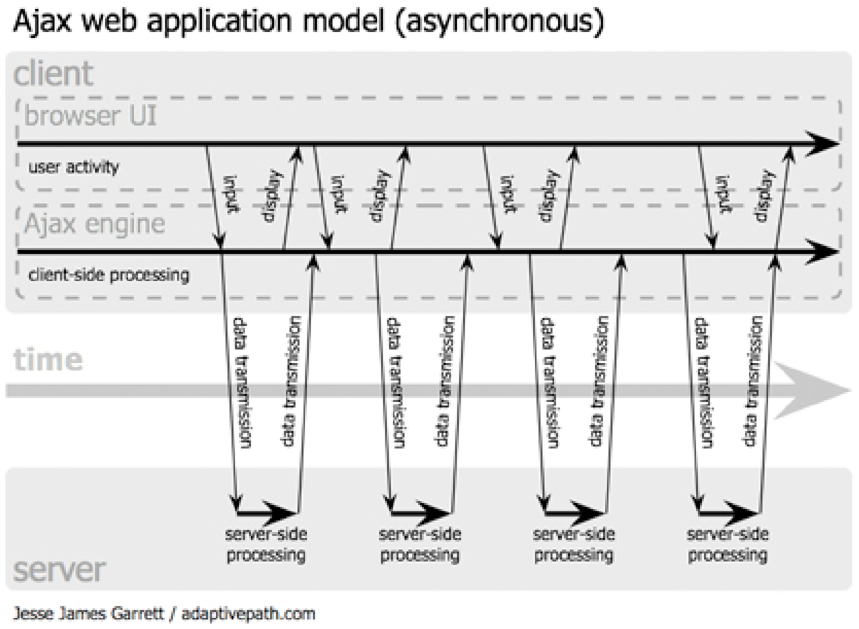
- We want pages that can load content at different points in time, depending on user interaction
    - Without reloading the entire page
    - Without showing another embedded webpage
    - By using web standards (for cross-browser compatibility)

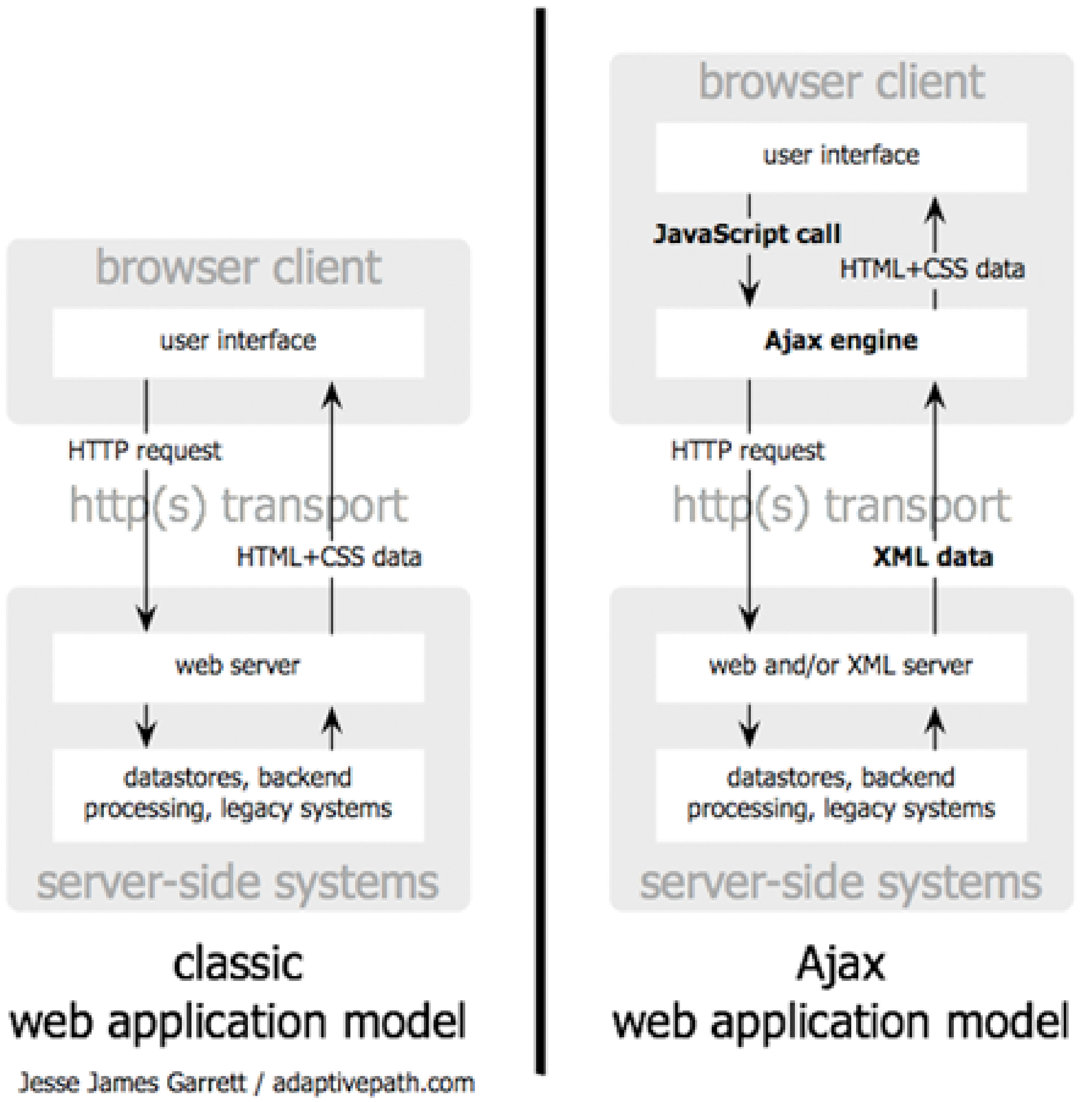
- Reloading the entire page for a small change in data is inefficient

AJAX

- Asynchronous JavaScript and XML (AJAX) is somewhat of a misnomer
- Today, XML and XSLT is not necessary. Instead most use JSON. When XHTML or HTML fragments are used, its known as Asynchronous HTML and HTTP (AHAH)
- The JavaScript XMLHttpRequest object is used for asynchronous communication






browser client
user interface
HTTP request
http(s) transport
HTML+CSS data
web server
datastores, backend processing, legacy systems
server-side systems
classic web application model
Jesse James Garrett / adaptivepath.com
browser client
user interface
JavaScript call
HTML+CSS data
Ajax engine
HTTP request
http(s) transport
XML data
web and/or XML server
datastores, backend processing, legacy systems
server-side systems
Ajax web application model

Using the XMLHttpRequest

- using native JavaScript object (XHR) - XMLHttpRequest

```
// Initialize the Ajax request (We ignore IE5, IE6 here)
var xhr = new XMLHttpRequest();

xhr.open('get', '/assignment2-gui/api/student.json');
// Track the state changes of the request
xhr.onreadystatechange = function () {
    // Ready state 4 means the request is done
    if (xhr.readyState === 4) {
        // 200 is a successful
        return if (xhr.status === 200) {
            document.getElementById("myDiv").innerHTML = xhr.responseText  // loading data into myDiv
        } else {
            alert('Error: ' + xhr.status);
            // An error occurred during the request
        }
    }
}

// Send the request to '/assignment2-gui/api/student.json'
xhr.send();
```

Using JQuery

- JQuery is a popular library for making cross-browser JS

```
$.ajax({
    type: "GET",
    url: "/assignment2-gui/api/student.json",
})
 .fail(function () {
        alert("error");
    })

.done(function (json) {
   $("#myDiv").html(json); // you probably want to do something to the JSON before putting it into HTML
});
```

```
// OR USING THE HIGH-LEVEL getJSON function
$.getJSON("/assignment2-gui/api/student.json", function (json) {
    $("#myDiv").html(json);  // you probably want to do something to the JSON before putting it into HTML
});
```

Challenges with AJAX

- Before HTML5 *History* interface, designing pages with back and forward for AJAX loaded content was difficult
- Dynamic updates to a page might happen very late on slow internet connections.
- Most web crawlers don’t execute JavaScript. Thus, AJAX loaded content is not indexed. Using AHAH can solve these as those fragments might be indexed.
- Same origin policy – Data from the same domain is only allowed to be loaded. To circumvent this, use JSONP or CORS, i.e. enable server-side response as well as browser-side JS to tell browser that data may come from another source

```
$.ajax({
    url: a_cross_domain_url,
    xhrFields: {
        withCredentials: true
}});
```

Geolocation

- HTML5 provides the Geolocation API to get user’s geographical location
- Location is considered private information and hence requires user permission before it’s shared
- Geolocation is more accurate for devices with GPS, like phones

```
var options = {
    enableHighAccuracy: true,
    timeout: 5000,
    maximumAge: 0
};

function getLocation() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(success, error, options);
    } else {
        x.innerHTML = "Geolocation is not supported by this browser.";
    }
}

function success (position){
    $("#myDiv").html("Latitude: " + position.coords.latitude + "<br>Longitude:" + position.coords.longitude);
}
function error () {… }
```

Maps

- Displaying Maps can be done using one of the many map displaying libraries or web services
- Some popular ones include:
  - Google Maps
  - Bing Maps
  - OpenStreetMaps/Leaflet.js

Using Maps

```
// after getting location… Static Google Maps Image API
function success (position){
   $("#myDiv").html("Latitude: " + position.coords.latitude + "<br>Longitude:" + position.coords.longitude);
   var img = new Image();
   var latlong = position.coords.latitude+ "," + position.coords.longitude;
   img.src = "http://maps.googleapis.com/maps/api/staticmap?center=" +latlong+ "&markers=" +latlong+
"&zoom=13&size=300x300&sensor=false";
   $("#map").html(img);
}
```

```
<script src="http://ecn.dev.virtualearth.net/mapcontrol/mapcontrol.ashx?v=6.3">
…
// after getting location… Using Bing Maps
function success(position) {
   $("#myDiv").html("Latitude: " + position.coords.latitude + "<br>Longitude:" + position.coords.longitude);
   var LA = new VELatLong(position.coords.latitude, position.coords.longitude);
   //create a new instance of VEMap
   map = new VEMap('mapContainer');
   //set options and load map.
   map.LoadMap(LA,
      4,
      VEMapStyle.Road,
      false,           //map view is displayed as a fixed map
      VEMapMode.Mode2D,    //VEMapMode.Mode2D. or VEMapMode.Mode3D.
      true,           //show switch on the map
      1             //tile buffer to use when loading map.
   );
   //create and add pushpin.
   pinPoint = map.GetCenter();
   pinPixel = map.LatLongToPixel(pinPoint);
   map.AddPushpin(pinPoint);
}
```

Concerns

- By using a 3rd party mapping service, you are giving away the user location to the 3rd party
- You may want to load maps and then point the client location in browser. Use leaflet.js or another mapping library

```
<link rel="stylesheet" href="http://cdn.leafletjs.com/leaflet-0.6.4/leaflet.css" />
<script src="http://cdn.leafletjs.com/leaflet-0.6.4/leaflet.js"></script>
<div id="map"></div>
<script>
var map = L.map('map').setView([position.coords.latitude, position.coords.longitude], 13);
var marker = L.marker([position.coords.latitude, position.coords.longitude]).addTo(map);
</script>
```

- Using device sensors is battery consuming
- To use device sensors with Google Maps API:

```
<script type="text/javascript" src="http://maps.googleapis.com/maps/api/js?sensor=true"></script>
<script>
var oslo = new google.maps.LatLng(59.913869, 10.752245);
map.setCenter(oslo);
</script>
```

Resources

- MDN – using geolocation
  - https://developer.mozilla.org/en-US/docs/WebAPI/Using_geolocation
- Jquery – Learn AJAX
  - http://learn.jquery.com/ajax/
- AHAH – Asynchronous HTML and HTTP
  - http://microformats.org/wiki/rest/ahah
- Google Maps API
  - https://developers.google.com/maps/articles/geolocation