IMPORTANT NOTICE TO STUDENTS

These slides are **NOT** to be used as a replacement for student notes.
These **slides** are sometimes **vague and incomplete on purpose** to spark class discussions

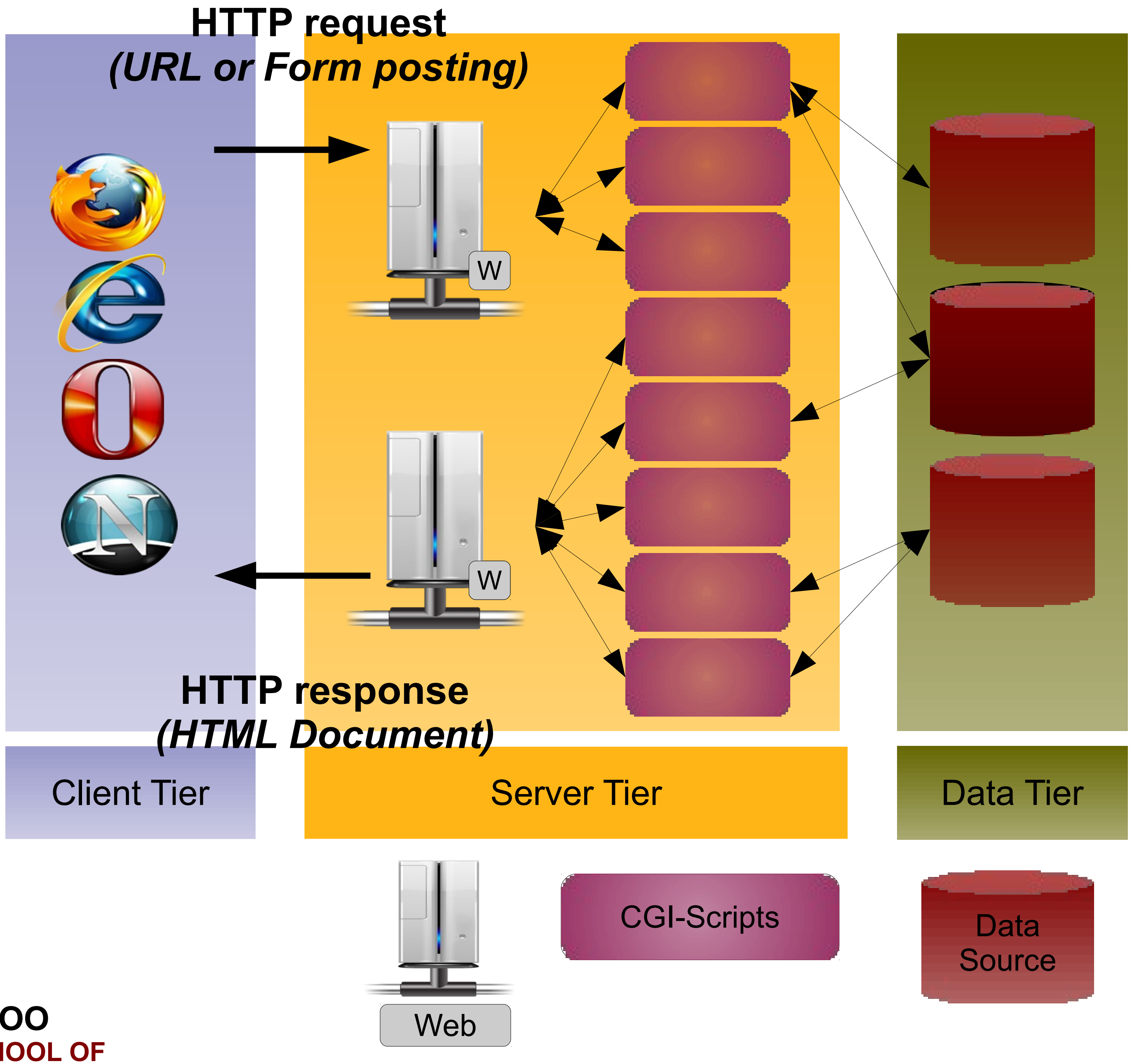
Google Web Toolkit (GWT)

"Architectural Impact on Web Applications"

CS 446/646 ECE452
Jun 8th, 2011

WATERLOO
CHERITON SCHOOL OF
COMPUTER SCIENCE

First Generation



First Generation

Shortcomings

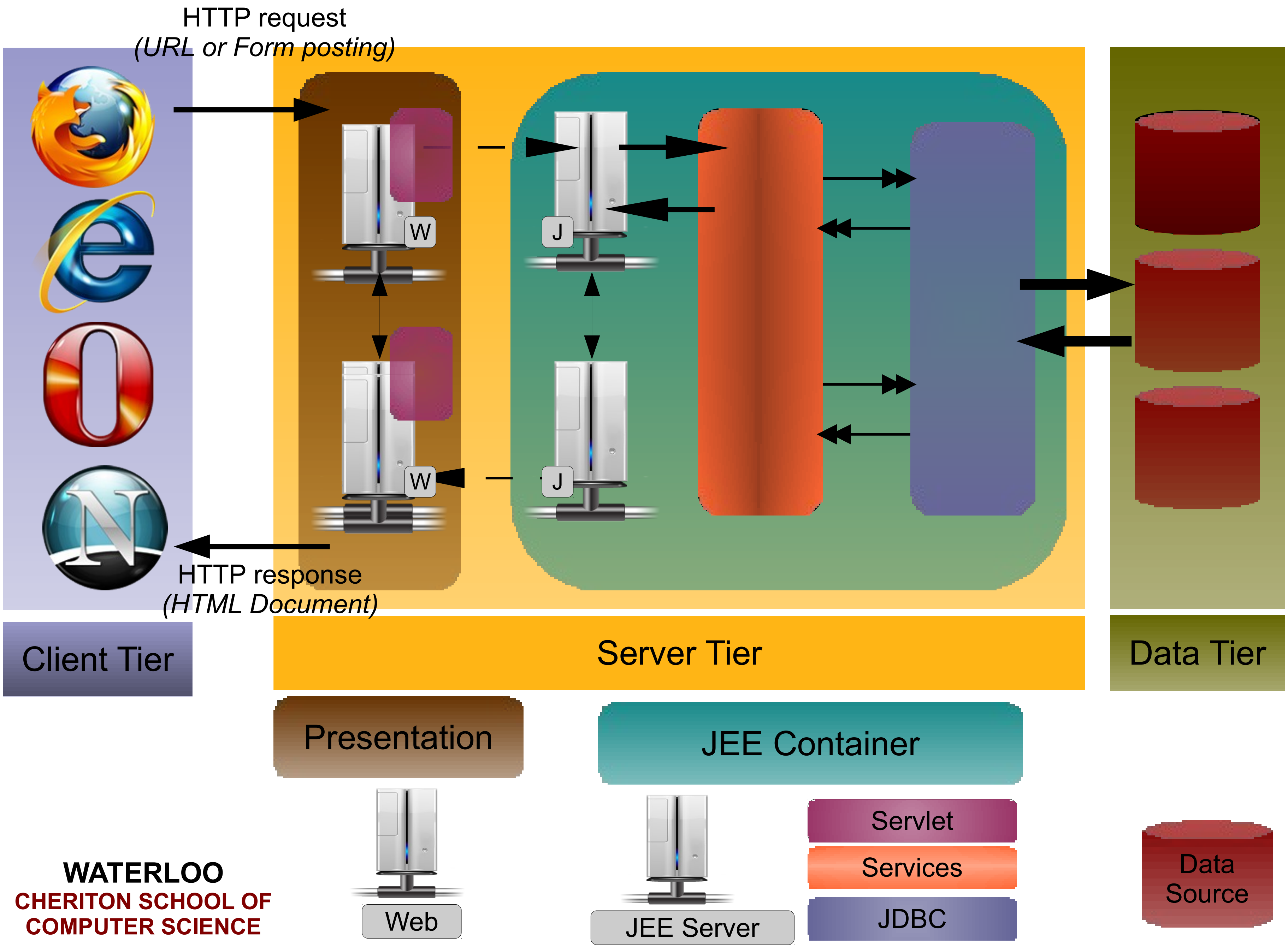
- lack of a coherent architectural model
  - development = adhoc composition of scripts
  - **limited** interaction between scripts
- evolution
  - how do you evolve scripts?
  - written in different languages
- data/information sharing
  - has to be via an external repository
  - difficult to control transactions

First Generation

Shortcomings

- security
  - CGI based scripts still a nightmare for admins
  - scripts are executed in native environment
    - server execution environment is directly exposed
    - vulnerabilities in server code are also exposed
- throughput
  - a script launches a process – ***so what?***
- tight coupling
  - each view is coupled to its corresponding script

Second Generation



WATERLOO
CHERITON SCHOOL OF COMPUTER SCIENCE

Second Generation

Shortcomings

- server focused
  - almost all improvements are on the server side
- client tier
  - still based on primitives
    - HTML, javascript, CSS etc.
  - *not dynamic*
- request-response cycle
  - worse than first generation ???

GWT

SG + AJAX = GWT

Not Exactly

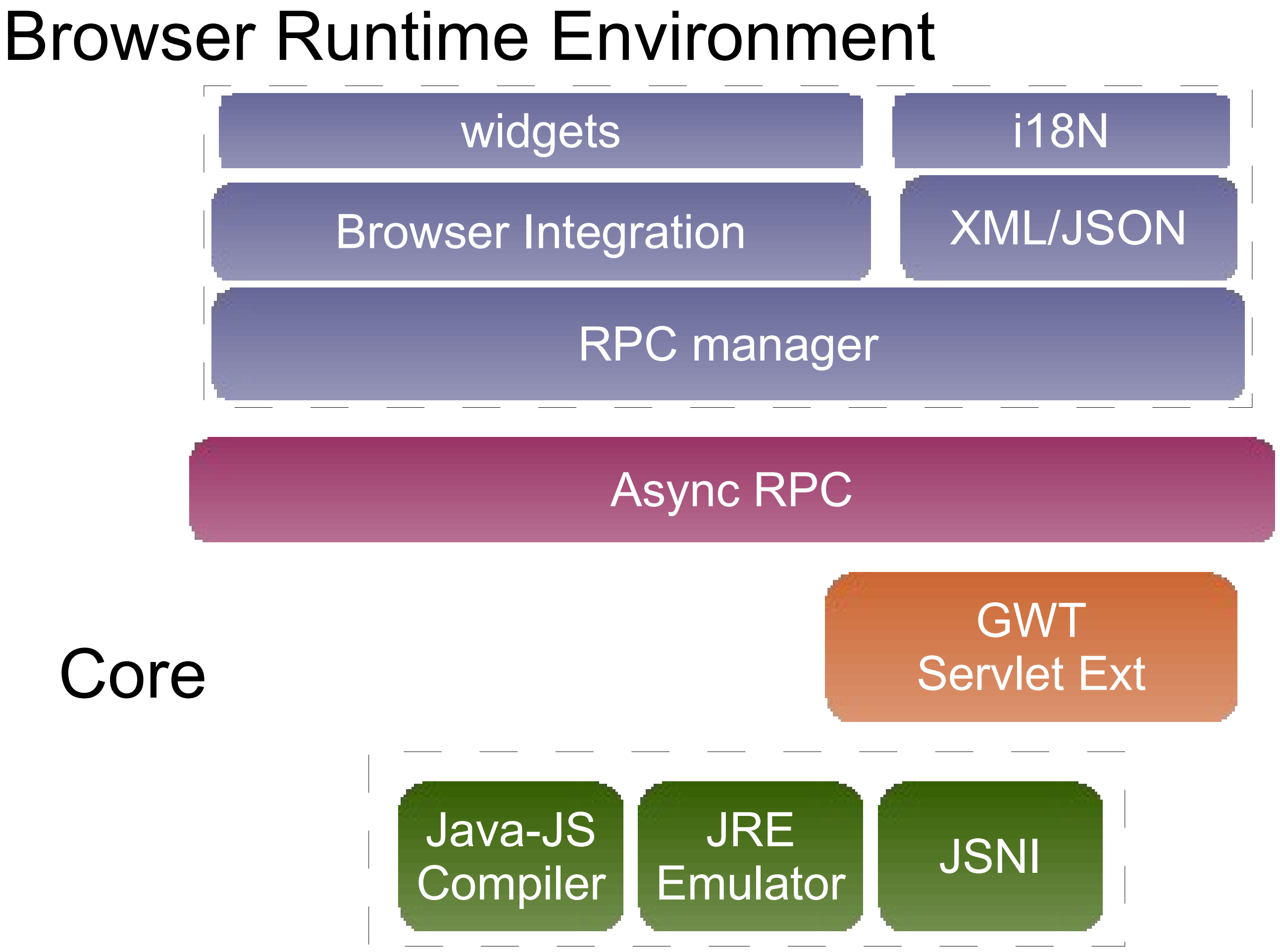
AJAX: Asynchronous JavaScript and XML

GWT

GWT is a lot more than that

- a paradigm shift away from
  - traditional (web application) building blocks
  - synchronous communication
- built on standards
  - integrates well with existing server applications & web services
- composite building blocks
  - HTML, javascript etc are low-level primitives
- separation of concerns

GWT Components



Java-JS Compiler

Converts Java to Javascript

- src-to-src compiler
- high level typed language to a script language
  - ***does this make sense???***
- JS code optimization
  - browser engines
  - size
  - security / obfuscation
  - localization

JRE Emulator

Emulates

- core Java classes in Javascript

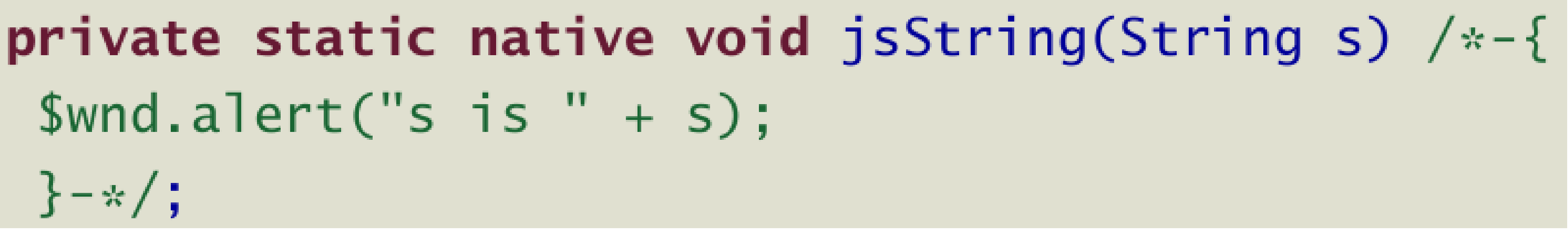
Composite building blocks

- allows for building composite building blocks
- client tier built on
  - composite building blocks rather than low level primitives

JSNI

Java Script Native Interface

- wrapper
  - for Javascript inside Java code
- extension point
  - for integration with non GWT client components



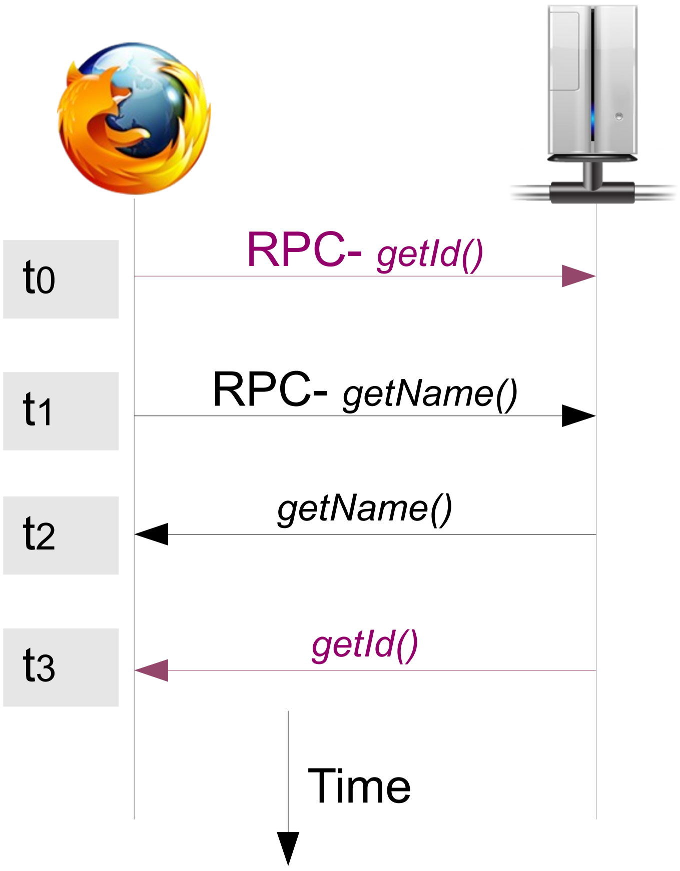
```
private static native void jsString(String s) /*-{
 $wnd.alert("s is " + s);
 }-*/;
```

Have we seen of this before?

RPC

Remote Procedure Call

- replaces HTTP
  - for communication after app boot
- asynchronous – ***why?***
  - breaks the request-response cycle
- supports various protocols
  - Ajax, JSON, GWT



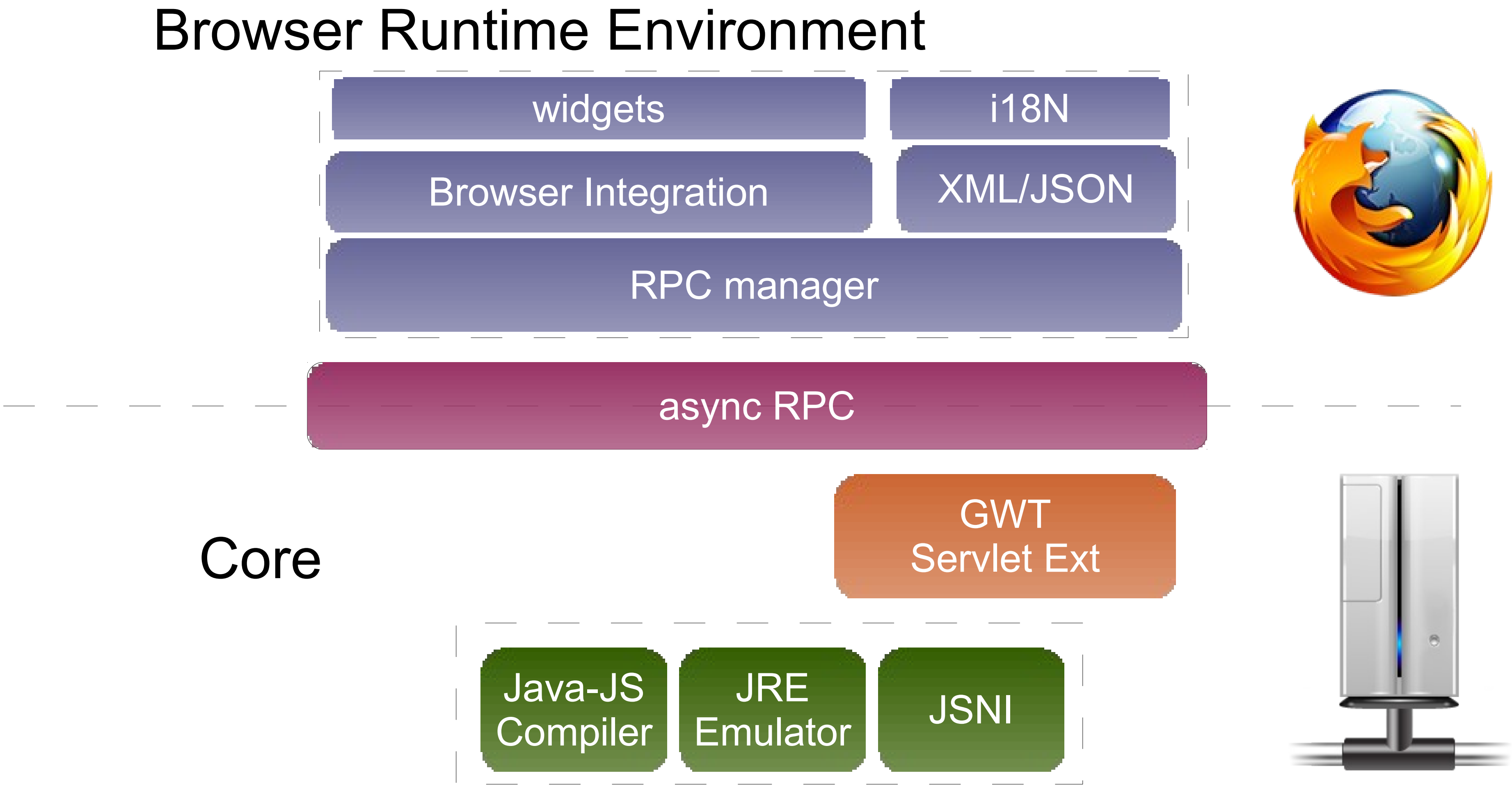
WATERLOO
CHERITON SCHOOL OF
COMPUTER SCIENCE

Servlet Extension

Extension of JEE Servlet

- integration point with JEE application
  - get all the JEE benefits for free
- server component
  - facade for business functionality
- evolution
  - highly flexible

GWT Components



Browser Runtime Environment

widgets

i18N

Browser Integration

XML/JSON

RPC manager

async RPC

Core

GWT Servlet Ext

Java-JS Compiler

JRE Emulator

JSNI

Impact

Scalability

Reusability

Interoperability

Design by contract

Evolution

Development

Scalability

Improvement in server performance

- (near) stateless servers
  - client tier components truly reside in client tier
  - previously
    - the state was maintained on the presentation tier
    - the view was rendered on the client tier
- optimized communication strategy
  - via aggregation of control/data exchange
  - decrease in server load
  - better bandwidth usage

Reusability

Application

- single code base to support
  - multiple browser engines
  - internationalization
    - i18N versions of the applications
- application broken over reusable modules

Design & development

- OOD
  - ***what benefits do we get from OOD?***

Interoperability

Integration / Extension points

- Javascript native interface (JSNI)
  - a layer of abstraction for integrating Javascript
    - third party & legacy Javascript libraries
- server side integration
  - servlet extension
    - plugs into the JEE platform
    - also possible for other platforms
  - mashups
    - use of diverse web services

Design by Contract

Client tier standardization

- browser runtime environment (BRE)
  - client code has to abide by the BRE interface
  - similar enhancement that JEE brought to server tier
  - ***isn't that strong coupling between GWT and an application client code?***
    - preserves the architectural integrity

Evolution

Organic growth

- OOA & OOD
  - ***what does this buy us?***
  - OO Javascript

Rich Internet Applications (RIA)

- HTML & HTTP as the basic building primitive
- prevails where others failed
  - Java applet, ActiveX, Adobe flex

Development

Java based Development

- well established
  - development tools & environments
    - IDEs, profiling
  - testing strategies
- skill-set standardization
  - Java developers already exist