Distributed Applications

Silvia Llorente

Jaime Delgado

Distributed Multimedia Applications Group
Departament d'Arquitectura de Computadors

(*) Some slides come from other sources

Web Applications

- Implement a remote application that can be accessed using a browser

- The ones proposed in the lab
  - Based on Servlets and Java Server Pages. Originally defined by Sun, now from Oracle

- This year you can implement them in other languages
  - Python, Javascript, php, other
  - They must provide the features defined in the documentation

Static Web Applications

- HTML pages
- Images
- Other static resources
  - Text document
  - Pdf files
  - Other
- Elements are always the same
  - Good for caching (proxy, browser, etc.)

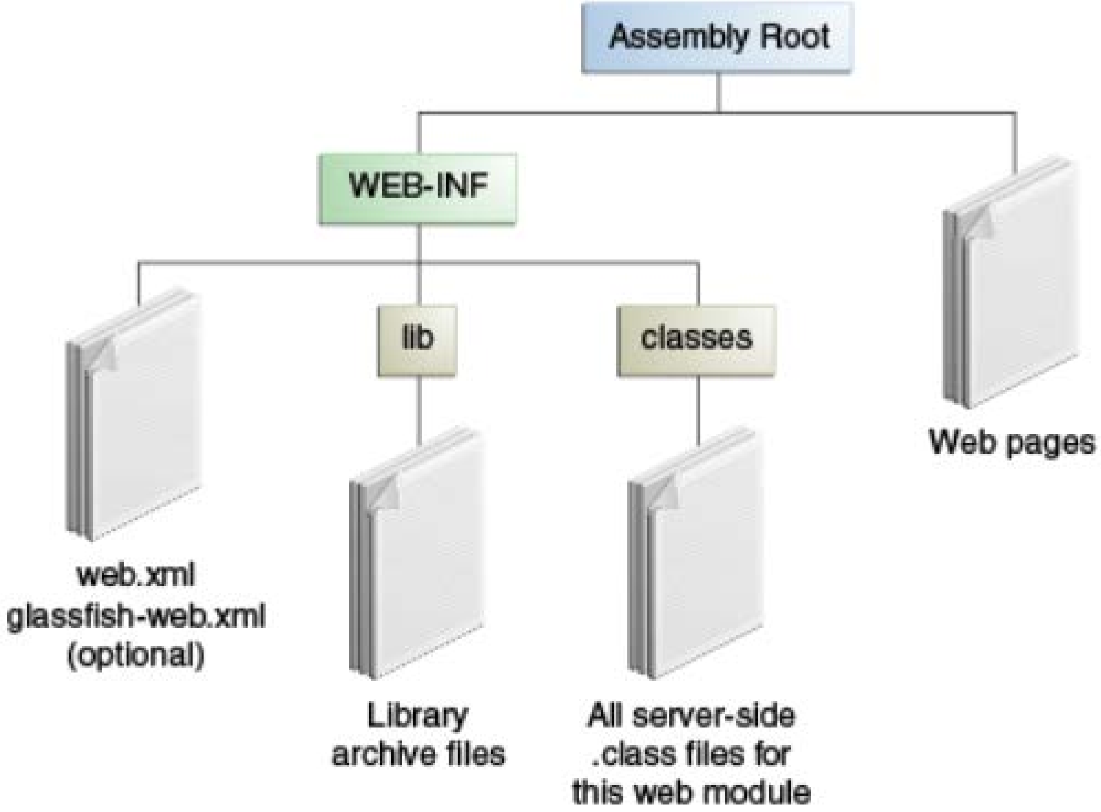
Dynamic Web Applications

- Use of some programming language to automatically generate HTML code getting to the browser
  - Oracle: Java Server Pages, Servlets
  - Python
  - Javascript
  - Microsoft: ASP
  - PHP
  - Executable files (.exe, scripts, etc.)

- Databases or external services can be used.

- Elements are generated dynamically
  - Static elements may also be used

Structure of installed J2EE Web Applications



- Root directory
  - Static HTML pages
  - Java Server Pages (jsp)
  - Images or other files to be published (pdf,doc's, etc.)
  - Allows subdirectories
  - WEB-INF

Web App's Structure Source: https://javaee.github.io/tutorial/packaging003.html

Structure (continued)

- WEB-INF directory
  - Special directory of the web application, contains configuration information
  - Classes
    - Contains java classes (.class files) used by the application
  - Lib
    - Contains java libraries (.jar files) used by the application
  - web.xml
    - Web application configuration file
    - Latest J2EE use Java Annotations
    - Ref:
    - https://docs.oracle.com/javase/tutorial/java/annotations/basics.html

web.xml simple file (J2EE 8)

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
   http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
   version="4.0">
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
</web-app>
```

XML Schema

Reference: https://javaee.github.io/servlet-spec/downloads/servlet-4.0/servlet-4_0_FINAL.pdf

Servlets

- Servlet
  - Java class dynamically loaded by server to add functionality
  - Executed in a java virtual machine
  - Platform independent
  - Generate HTML code to be sent to the client (not recommended) or redirect to a different page (jsp, HTML, another servlet, etc.)

Servlet life cycle

1) Create and initialize servlet

2) Process web client requests

3) Destroy servlet, freeing memory and resources (*garbage collection*)

Servlets API

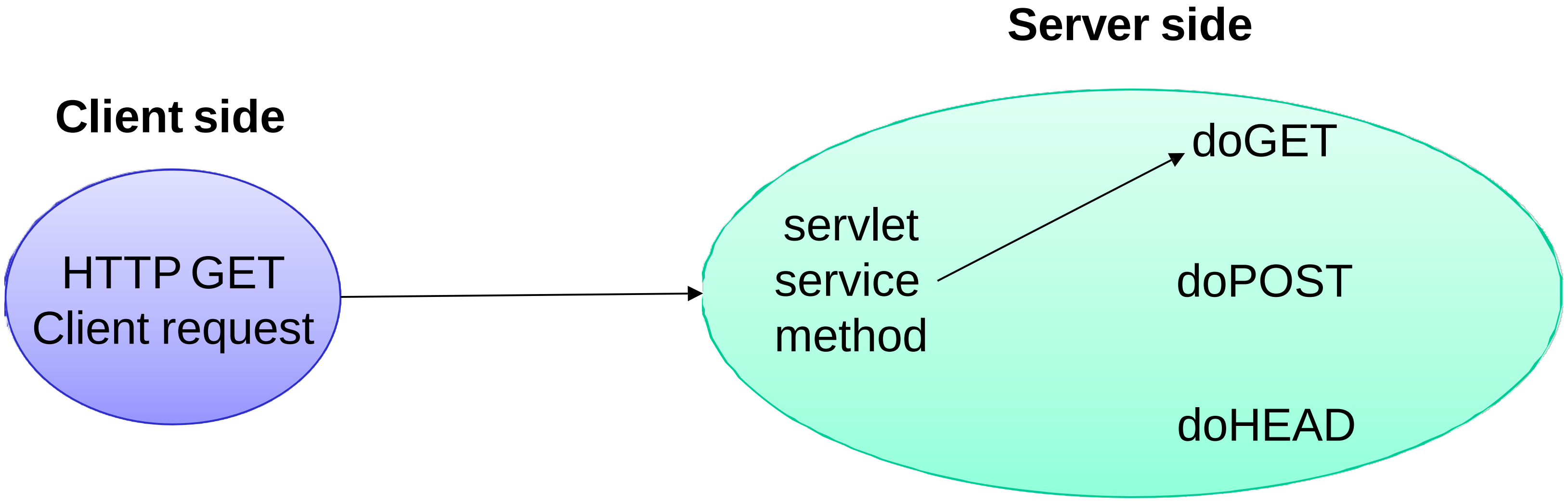
- 2 java packets
  - javax.servlet
  - javax.servlet.http

- Generic Servlets (javax.servlet)
  - Do not have main()
  - service() method is invoked when a request is received

HTTP Servlets

- HTTP Specific
- *service* method already implemented
  - Invokes corresponding HTTP method
  - doXXX (where XXX can be Get, Post, other HTTP methods)

HTTP Servlets: Example



- Client side makes a call to the servlet via its URL
- *service* method for servlet waits for requests
  - Invokes doGET
  - Servlet may implement several doXXX methods
  - When an HTTP method is not implemented, servlet returns an exception indicating that the method is not supported

Java Server Pages (JSP)

- Mix HTML with Java code
- Useful for interfaces, not command processing
- Specific tags to combine HTML code with Java
- Reference
  - http://www.exforsys.com/tutorials/jsp/jsp-introduction.html

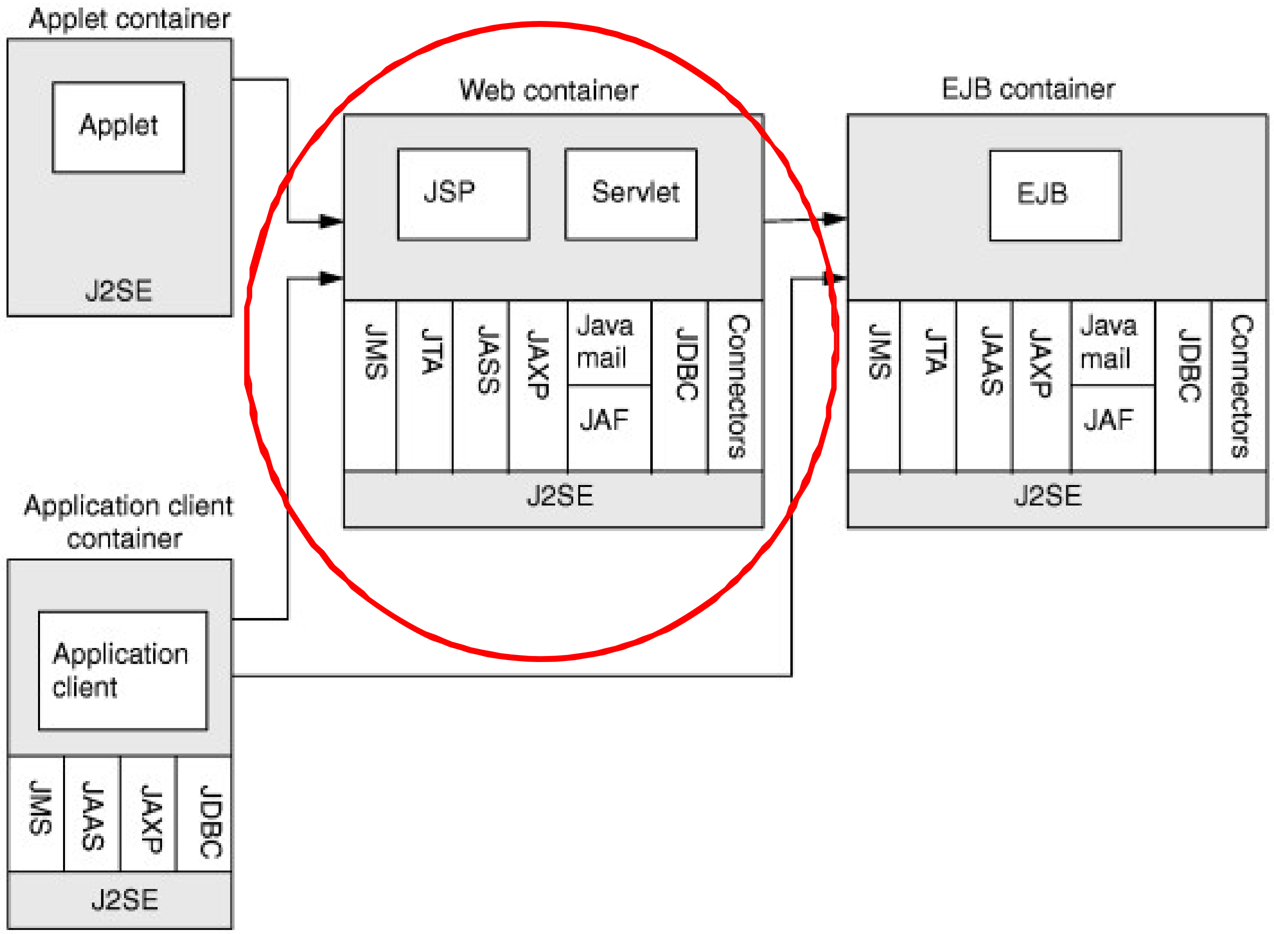
Servlets vs Java Server Pages

- Same functionality
- JSP easier to develop, but code is more confusing (mix of HTML and Java), useful for interfaces
- Server converts JSP into a servlet internally
  - Generates Java code for all jsp instructions, including HTML code
  - Compiles Java code
  - Calls the class generated and compiled when JSP is invoked
- JSP more difficult to debug, although IDE has improved

Application Servers

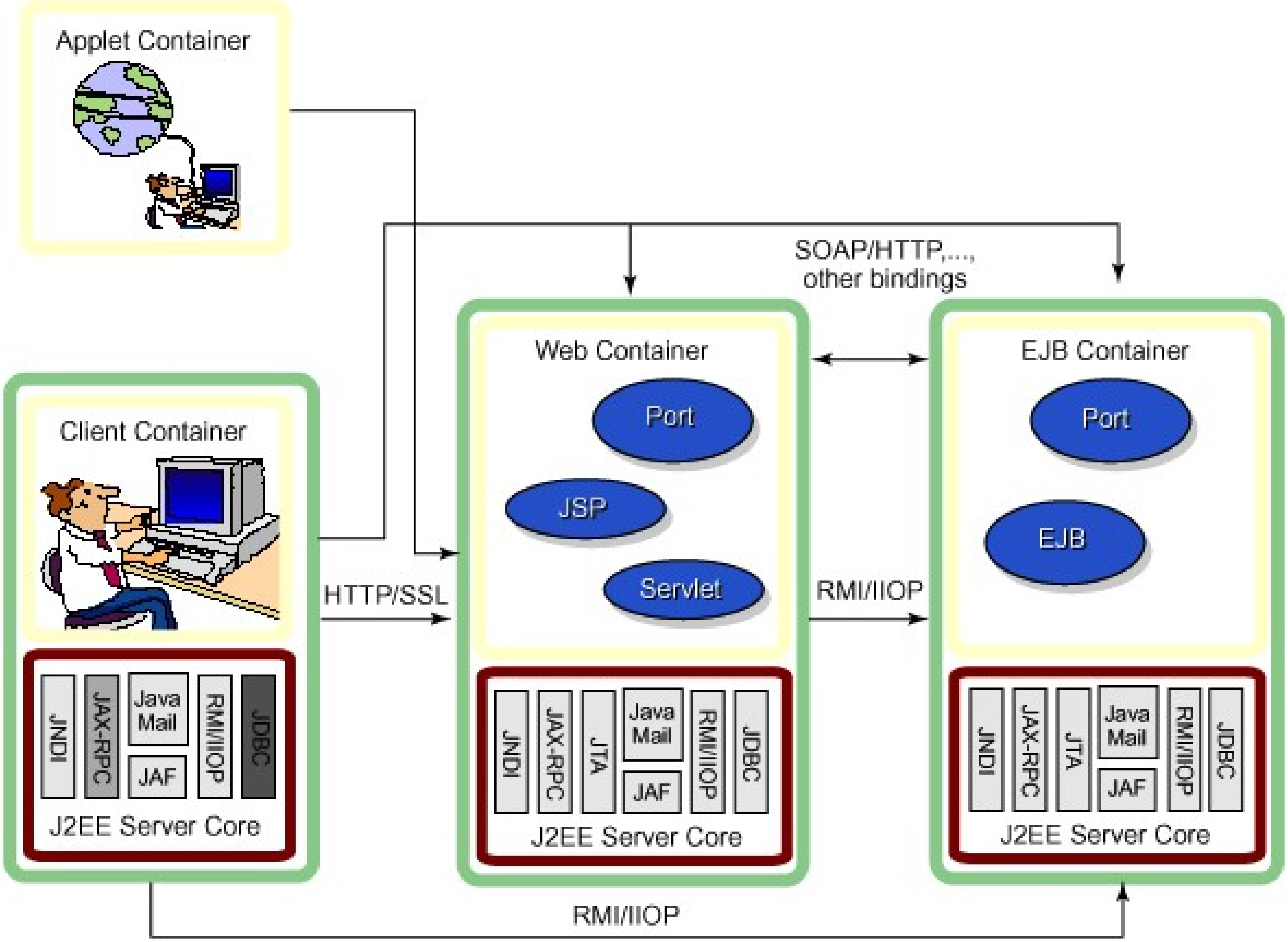
- The ones in the subject are based on J2EE model defined by Sun (now Oracle)
- Allow working with web applications, web services and business components (Java Beans, not used in class)
  - Complex applications, used by companies
- J2EE applications are portable between application servers, as they are developed in Java and with a complete structure
  - .war files, Web Application Archive
- Reference: J2EE compatible implementations https://www.oracle.com/java/technologies/compatibility-jsp.html

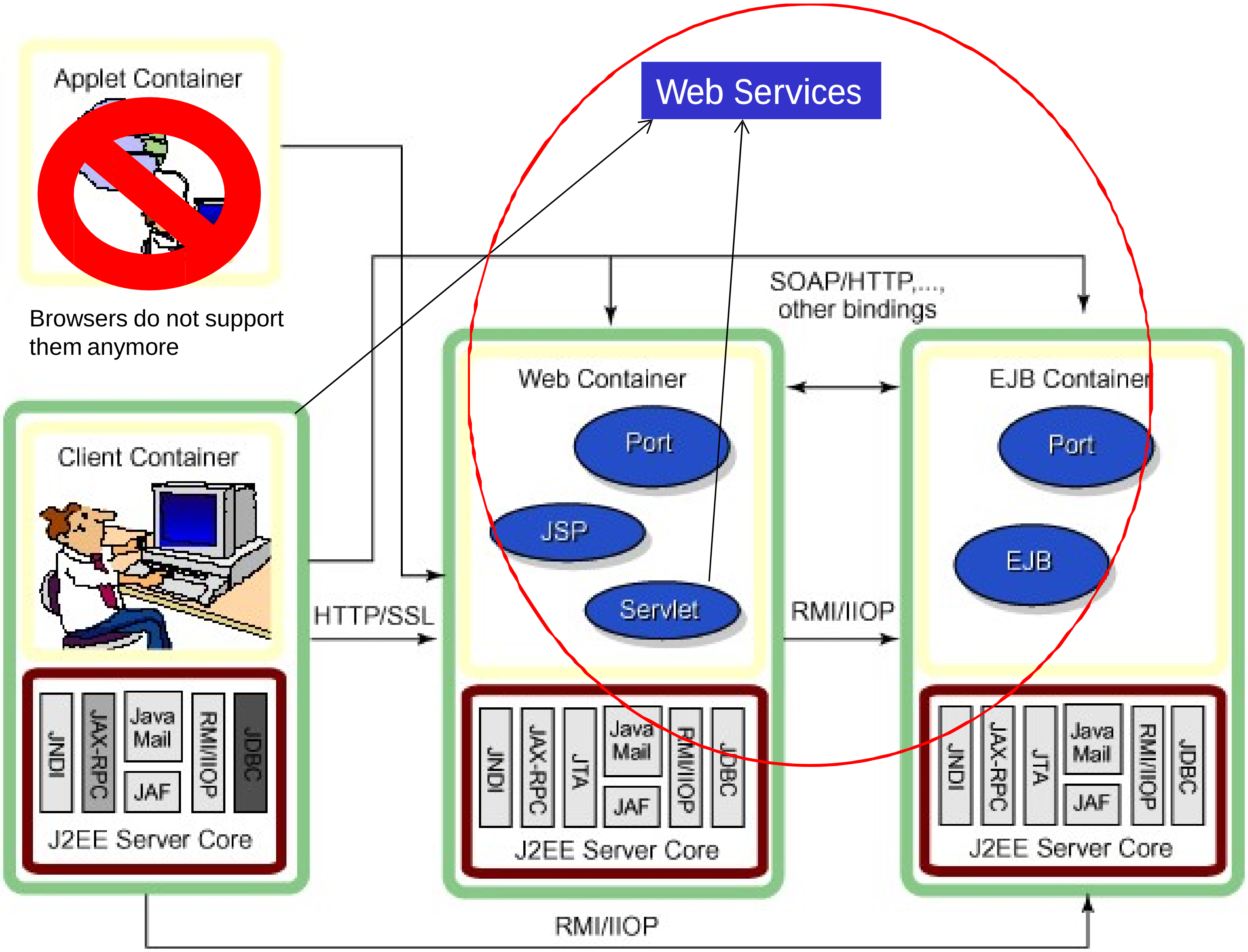
J2EE Application Server Architecture



J2SE = Java 2 Platform, Standard Edition
JAAS = Java Authentication and Authorization Service
JAF = JavaBeans Activation Framework
JAXP = Java API for XML Parsing

JDBC = Java Database Connectivity
JMS = Java Message Service
JSP = Java Server Page
JTA = Java Transaction API


Applet Container
SOAP/HTTP,...,
other bindings
Web Container
Port
JSP
Servlet
EJB Container
Port
EJB
Client Container
HTTP/SSL
RMI/IIOP
JNDI
JAX-RPC
Java Mail
JAF
RMI/IIOP
JDBC
J2EE Server Core
JNDI
JAX-RPC
JTA
Java Mail
JAF
RMI/IIOP
JDBC
J2EE Server Core
JNDI
JAX-RPC
JTA
Java Mail
JAF
RMI/IIOP
JDBC
J2EE Server Core
RMI/IIOP



Web Services

Browsers do not support them anymore

Other remote invocation methods

Glassfish

- J2EE application server offered by Eclipse
- We will use version 5
- References
  - Glassfish official page
    - https://projects.eclipse.org/projects/ee4j.glassfish/downloads

Apache Netbeans 12

- Currently supported by Apache
- Supports Java EE since version 11
- Official page:
  - https://netbeans.apache.org/

Distributed Applications

Silvia Llorente

Jaime Delgado

Distributed Multimedia Applications Group

Departament d'Arquitectura de Computadors

(*) Some slides come from other sources