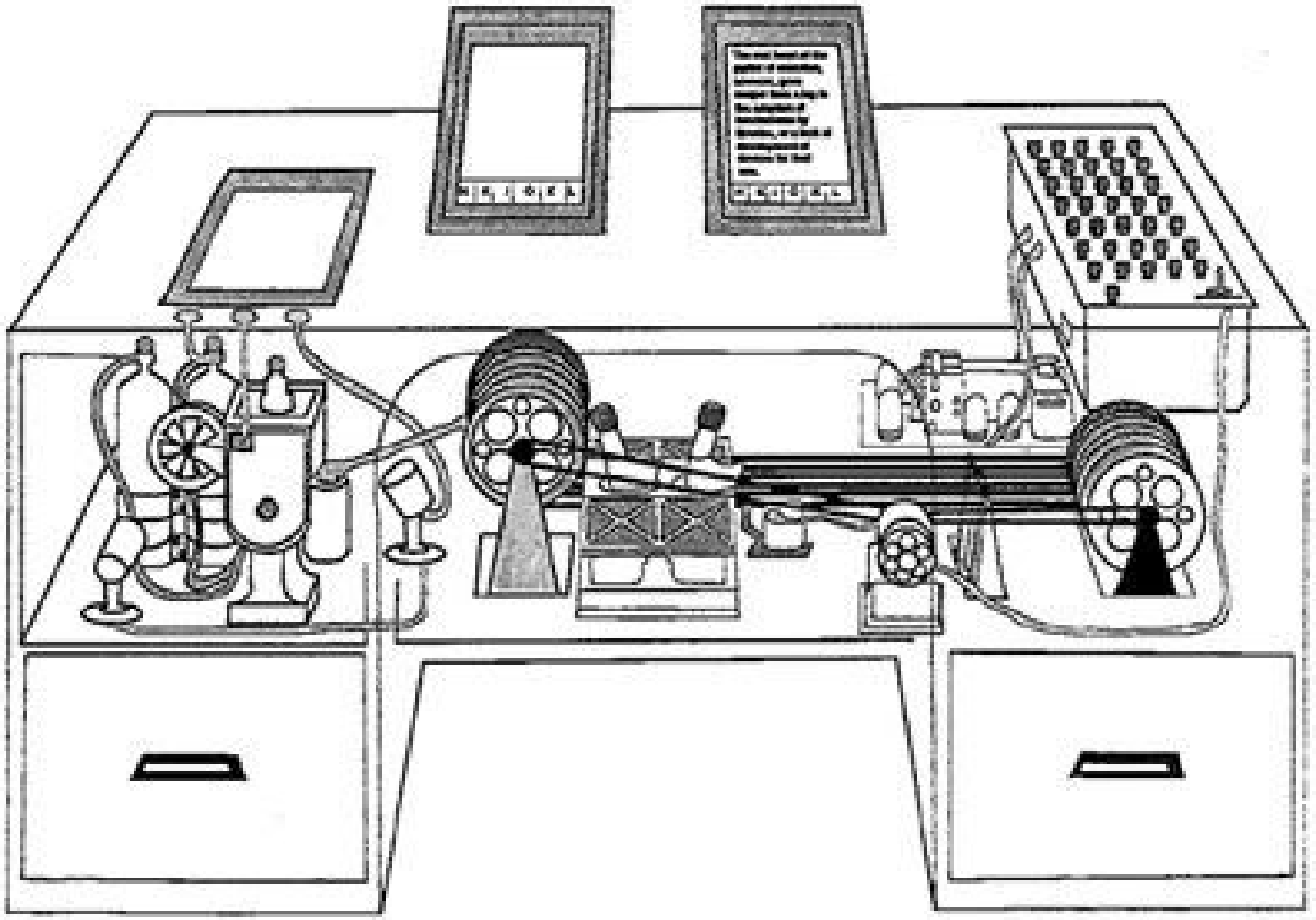
Java Enterprise Edition y Spring

Jaime A. Pavlich-Mariscal

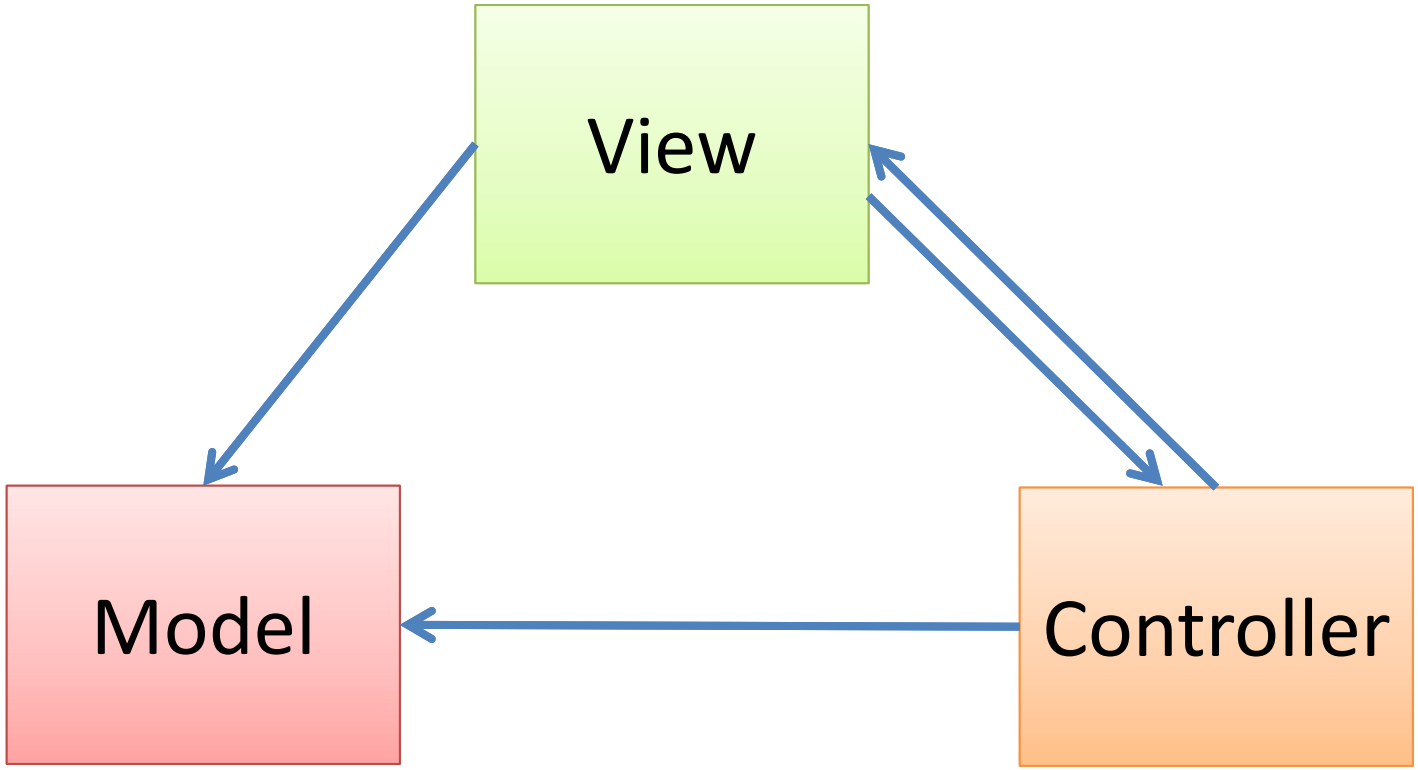
A little of history

1930s: Memex



Vannevar Bush. “As We May Think”. Atlantic Monthly. 1945

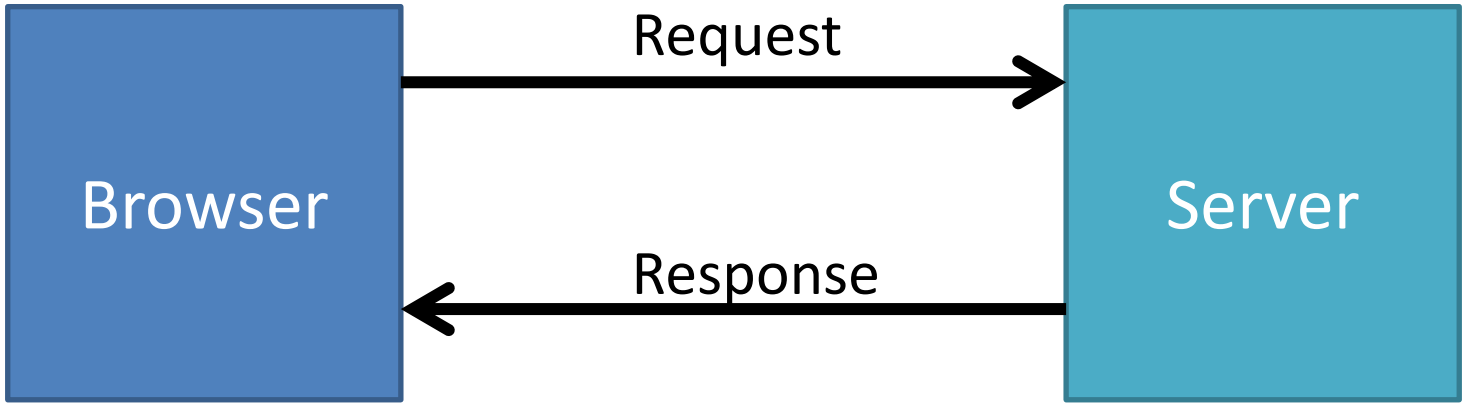
1988: Model-View-Controller



Krasner, Glenn E., and Stephen T. Pope. "A description of the model-view-controller user interface paradigm in the smalltalk-80 system." Journal of object oriented programming 1.3 (1988): 26-49.

1989: HTTP Standard

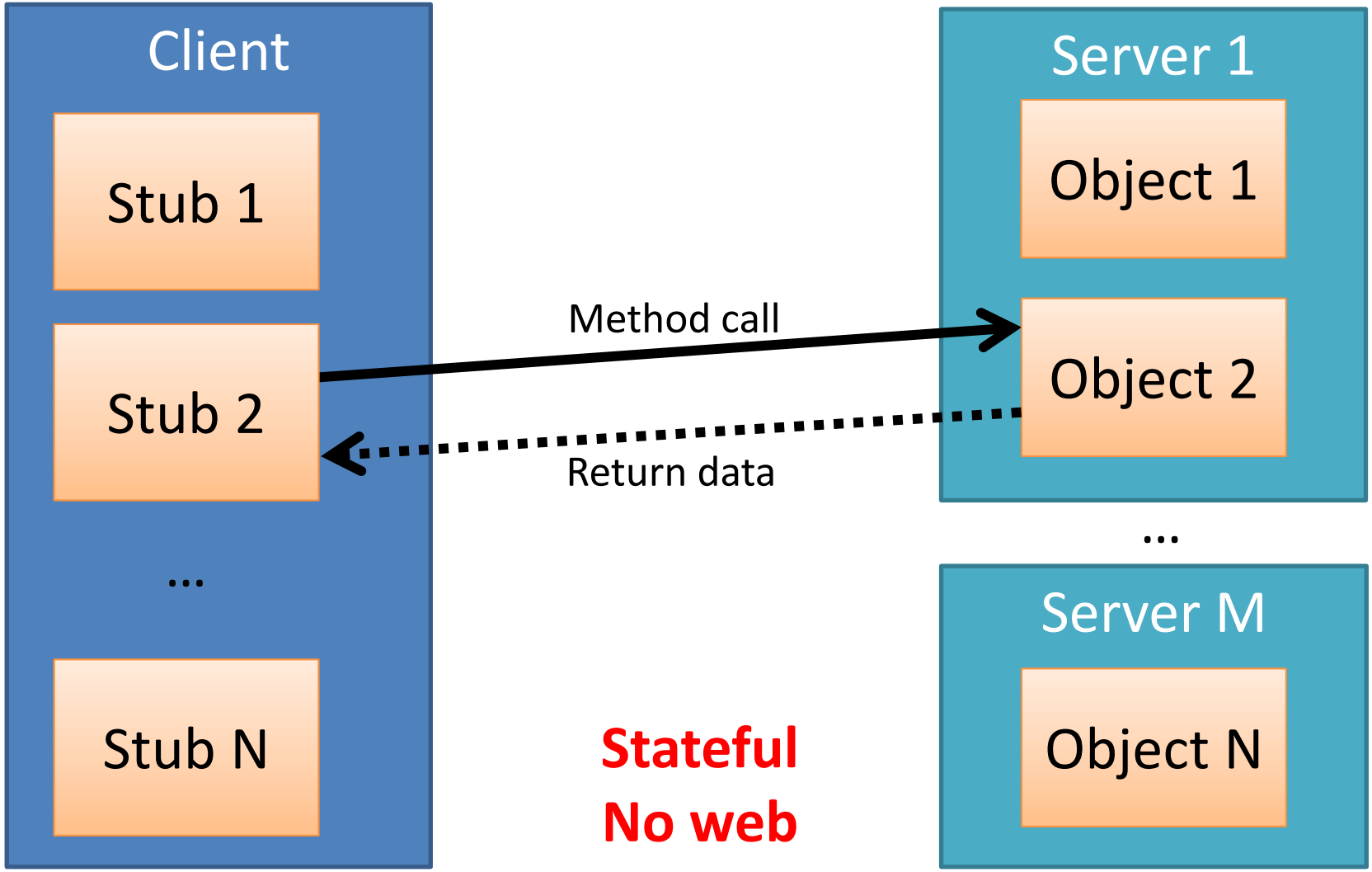
Online Hypertext



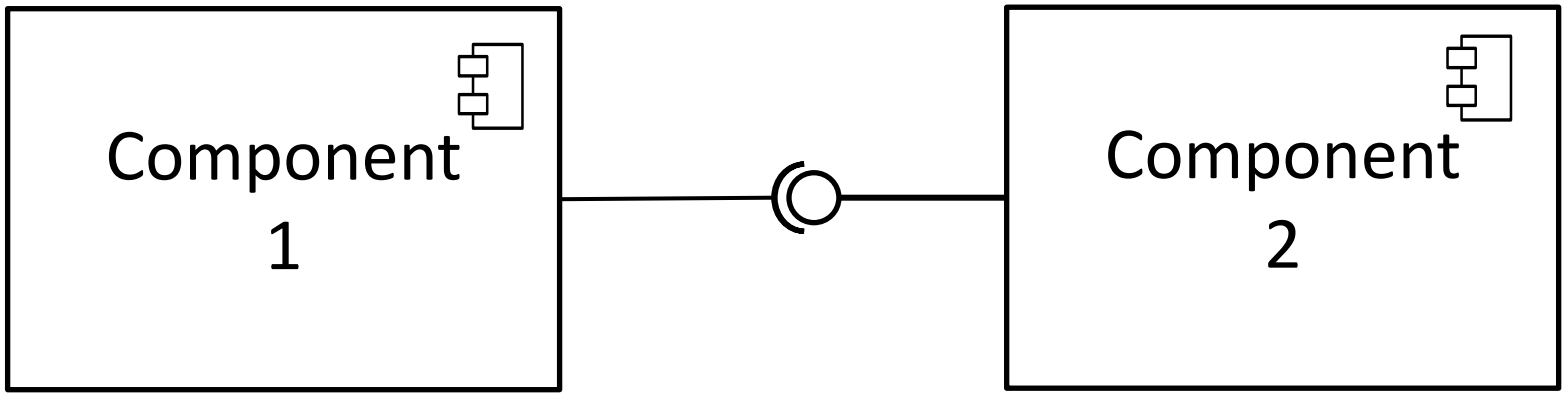
Stateless!!

1991: Corba 1.0

Distributed Objects



1998:Component-Based Software Engineering (CBSE)



Component software: beyond object-oriented programming. ACM Press/Addison-Wesley Publishing Co. New York, NY, USA ©1998. ISBN:0-201-17888-5

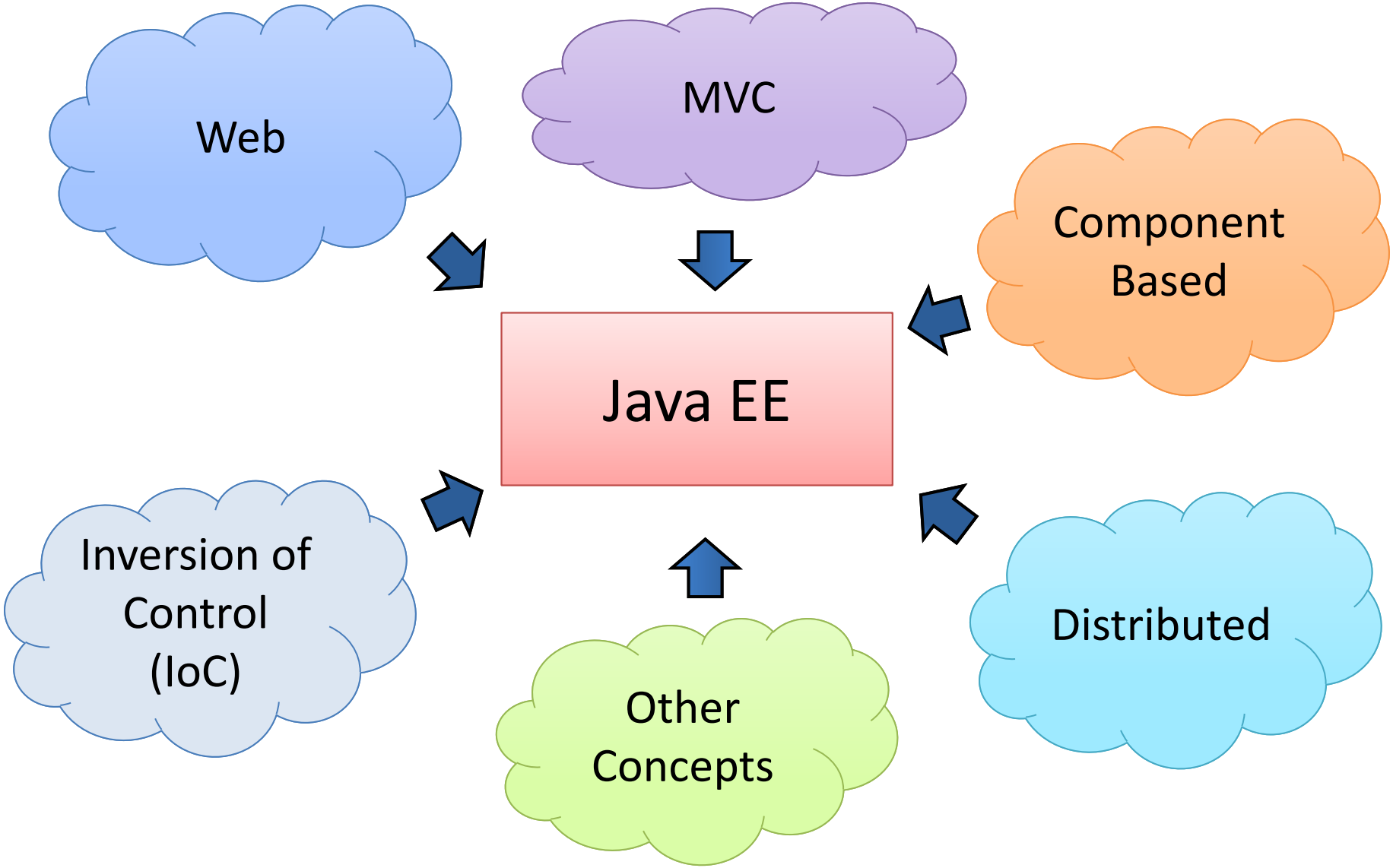
2004: Inversion of Control

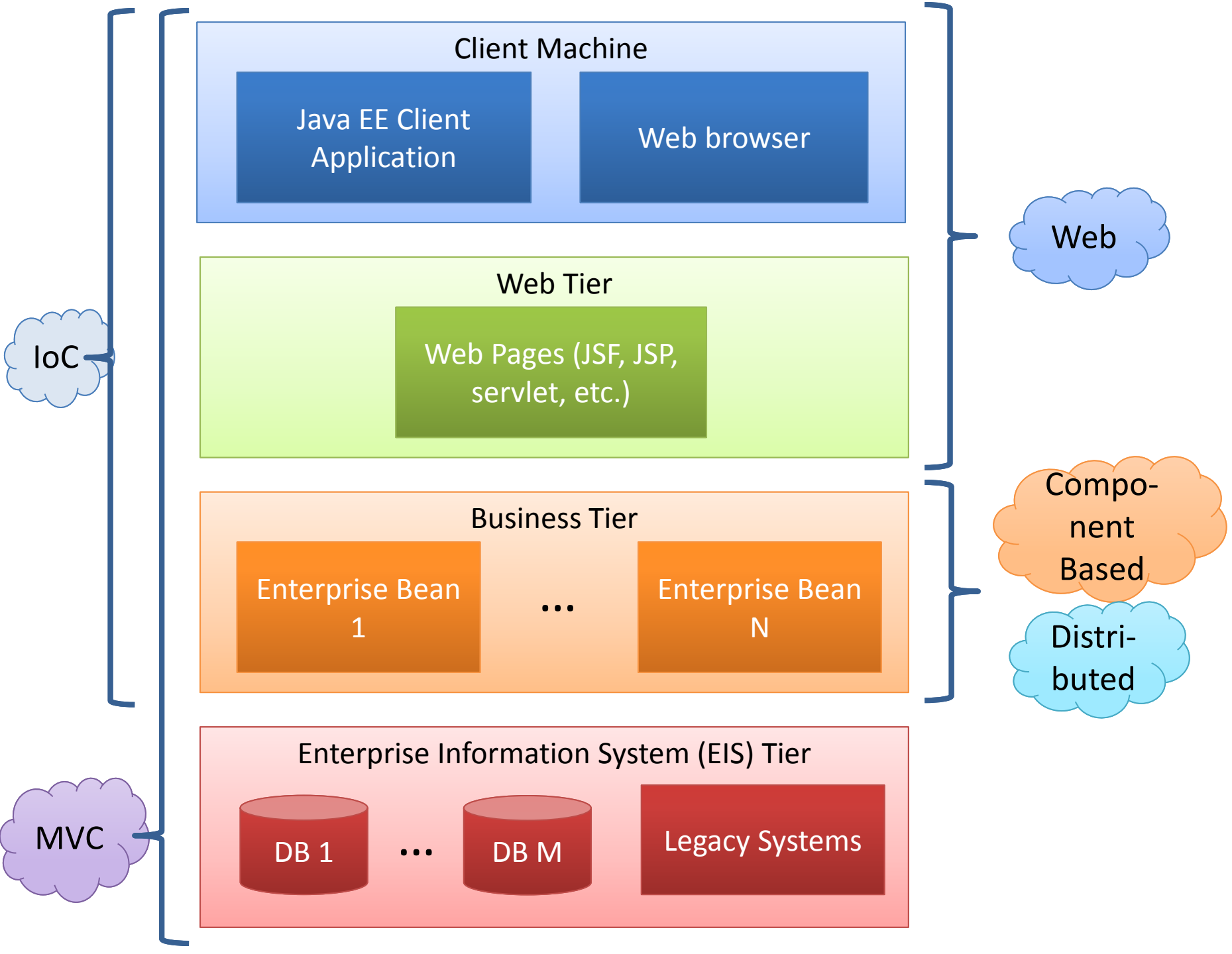
Libraries

Frameworks

http://martinfowler.com/articles/injection.html

Java EE & other similar frameworks




Client Machine
Java EE Client Application
Web browser
Web Tier
Web Pages (JSF, JSP, servlet, etc.)
Business Tier
Enterprise Bean 1
...
Enterprise Bean N
Enterprise Information System (EIS) Tier
DB 1
...
DB M
Legacy Systems
IoC
MVC
Web
Compo-nent Based
Distri-buted
https://docs.oracle.com/javaee/7/tutorial/overview003.htm

Java EE Specification

- http://www.oracle.com/technetwork/java/javaee/tech/index.html

What is the best way to learn it?

Java EE 7 Technologies

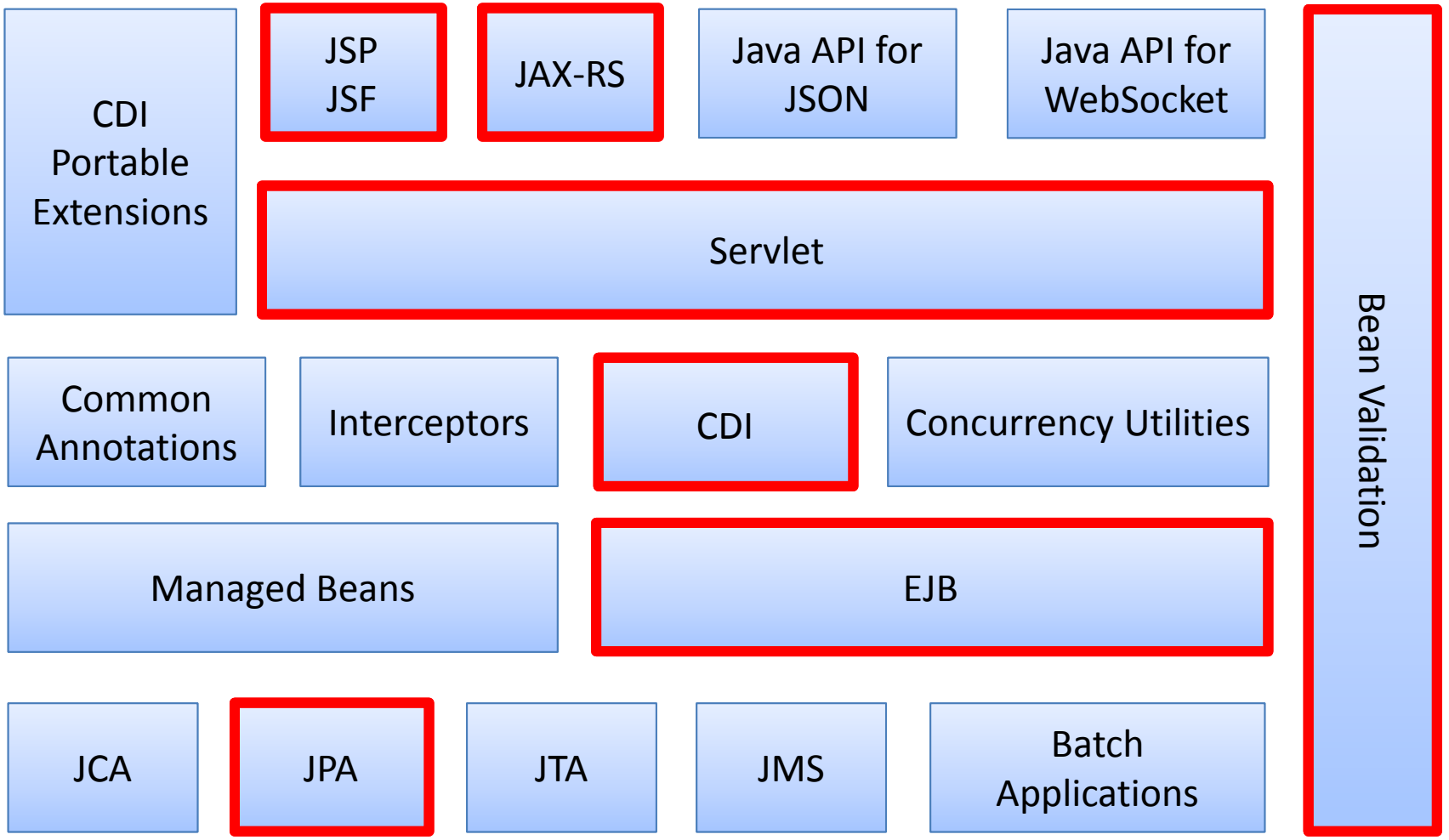
Learn more about the technologies that comprise the Java EE 7 platform using the specifications, and then apply them with the Java EE 7 SDK.

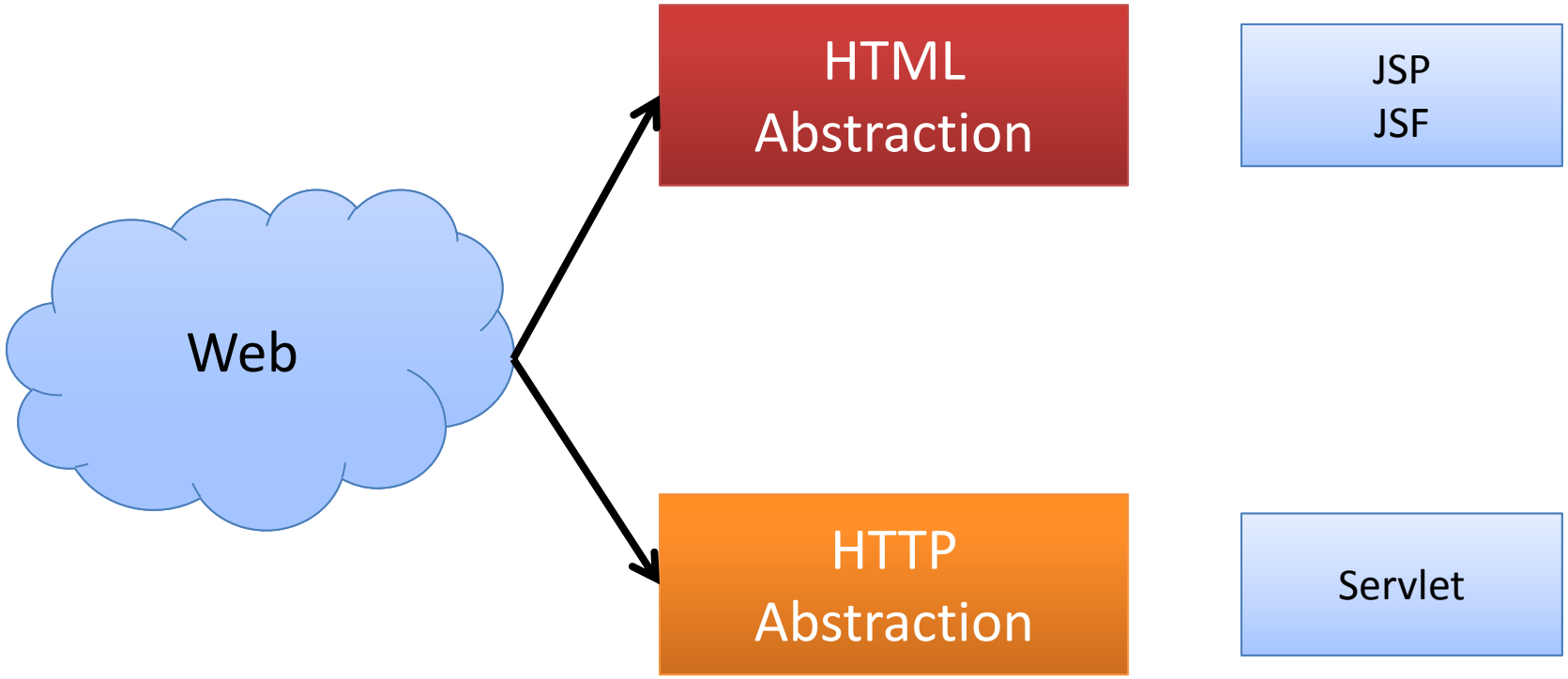
Specification downloads are the final releases. Please check the individual JSR pages for download updates such as maintenance releases.

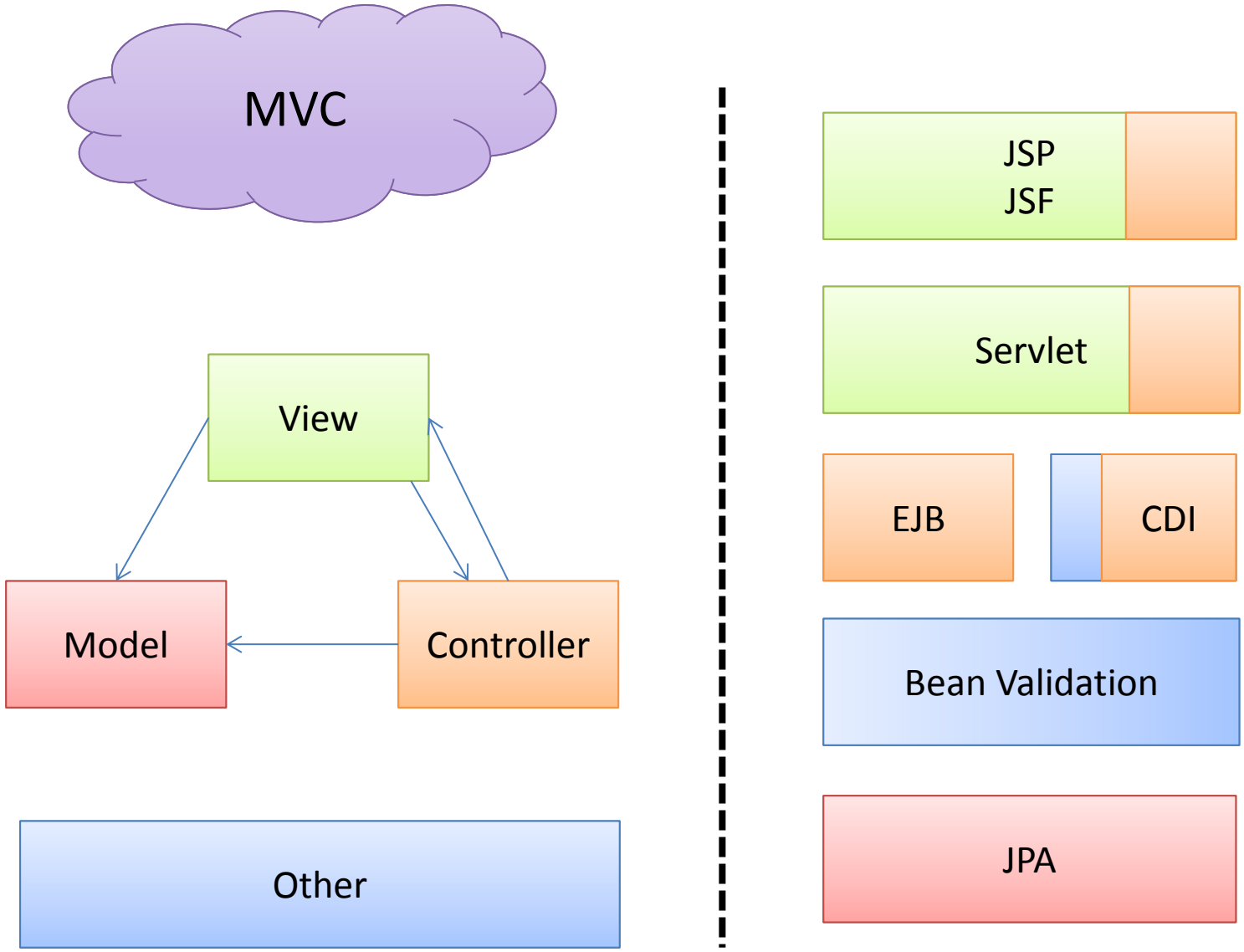
Java EE 7 Technologies			
Technologies	**JSR**	**Download**	**Web Profile**
Java EE Platform			
Java Platform, Enterprise Edition 7 (Java EE 7)	JSR 342	Download spec	
Web Application Technologies			
Java API for WebSocket	JSR 356	Download spec	✓
Java API for JSON Processing	JSR 353	Download spec	✓
Java Servlet 3.1	JSR 340	Download spec	✓
JavaServer Faces 2.2	JSR 344	Download spec	✓
Expression Language 3.0	JSR 341	Download spec	✓
JavaServer Pages 2.3	JSR 245	Download spec	✓
Standard Tag Library for JavaServer Pages (JSTL) 1.2	JSR 52	Download spec	✓
Enterprise Application Technologies			
Batch Applications for the Java Platform	JSR 352	Download spec	
Concurrency Utilities for Java EE 1.0	JSR 236	Download spec	
Contexts and Dependency Injection for Java 1.1	JSR 346	Download spec	✓
Dependency Injection for Java 1.0	JSR 330	Download spec	✓
Bean Validation 1.1	JSR 349	Download spec	✓
Enterprise JavaBeans 3.2	JSR 345	Download spec	✓
Interceptors 1.2 (Maintenance Release covered under JSR 318)	JSR 318	Download spec	✓
Java EE Connector Architecture 1.7	JSR 322	Download spec	
Java Persistence 2.1	JSR 338	Download spec	✓
Common Annotations for the Java Platform 1.2	JSR 250	Download spec	✓
Java Message Service API 2.0	JSR 343	Download spec	
Java Transaction API (JTA) 1.2	JSR 907	Download spec	✓
JavaMail 1.5	JSR 919	Download spec	
Web Services Technologies			
Java API for RESTful Web Services (JAX-RS) 2.0	JSR 339	Download spec	✓
Implementing Enterprise Web Services 1.3	JSR 109	Download spec	
Java API for XML-Based Web Services (JAX-WS) 2.2	JSR 224	Download spec	
Web Services Metadata for the Java Platform	JSR 181	Download spec	
Java API for XML-Based RPC (JAX-RPC) 1.1 (Optional)	JSR 101	Download spec	
Java APIs for XML Messaging 1.3	JSR 67	Download spec	
Java API for XML Registries (JAXR) 1.0	JSR 93	Download spec	
Management and Security Technologies			
Java Authentication Service Provider Interface for Containers 1.1	JSR 196	Download spec	
Java Authorization Contract for Containers 1.5	JSR 115	Download spec	
Java EE Application Deployment 1.2 (Optional)	JSR 88	Download spec	
J2EE Management 1.1	JSR 77	Download spec	
Debugging Support for Other Languages 1.0	JSR 45	Download spec	✓
Java EE-related Specs in Java SE			
Java Architecture for XML Binding (JAXB) 2.2	JSR 222	Download spec	
Java API for XML Processing (JAXP) 1.3	JSR 206	Download spec	
Java Database Connectivity 4.0	JSR 221	Download spec	
Java Management Extensions (JMX) 2.0	JSR 003	Download spec	
JavaBeans Activation Framework (JAF) 1.1	JSR 925	Download spec	
Streaming API for XML (StAX) 1.0	JSR 173	Download spec	

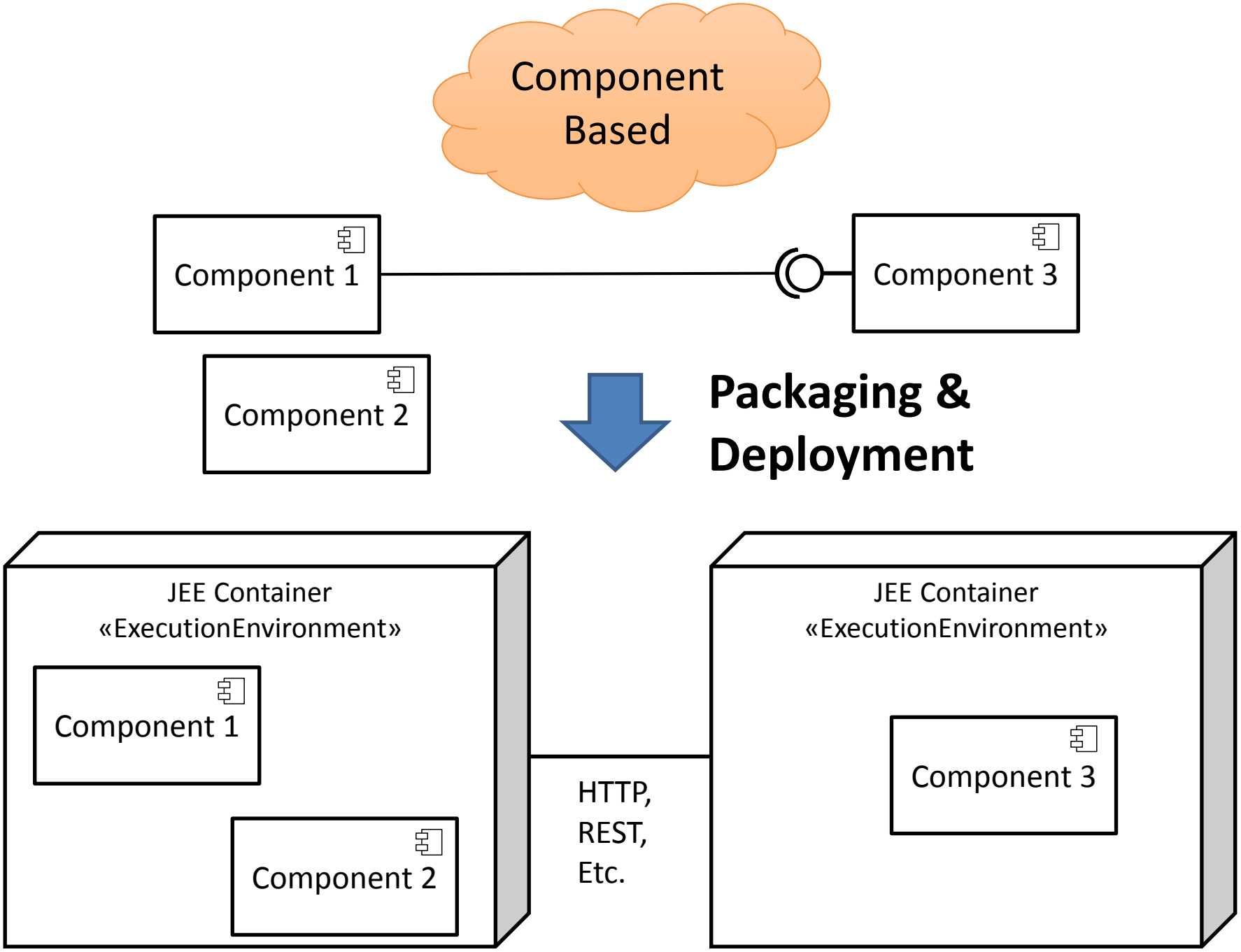
See Also:
Java EE 6 Technologies
Java EE 5 Technologies

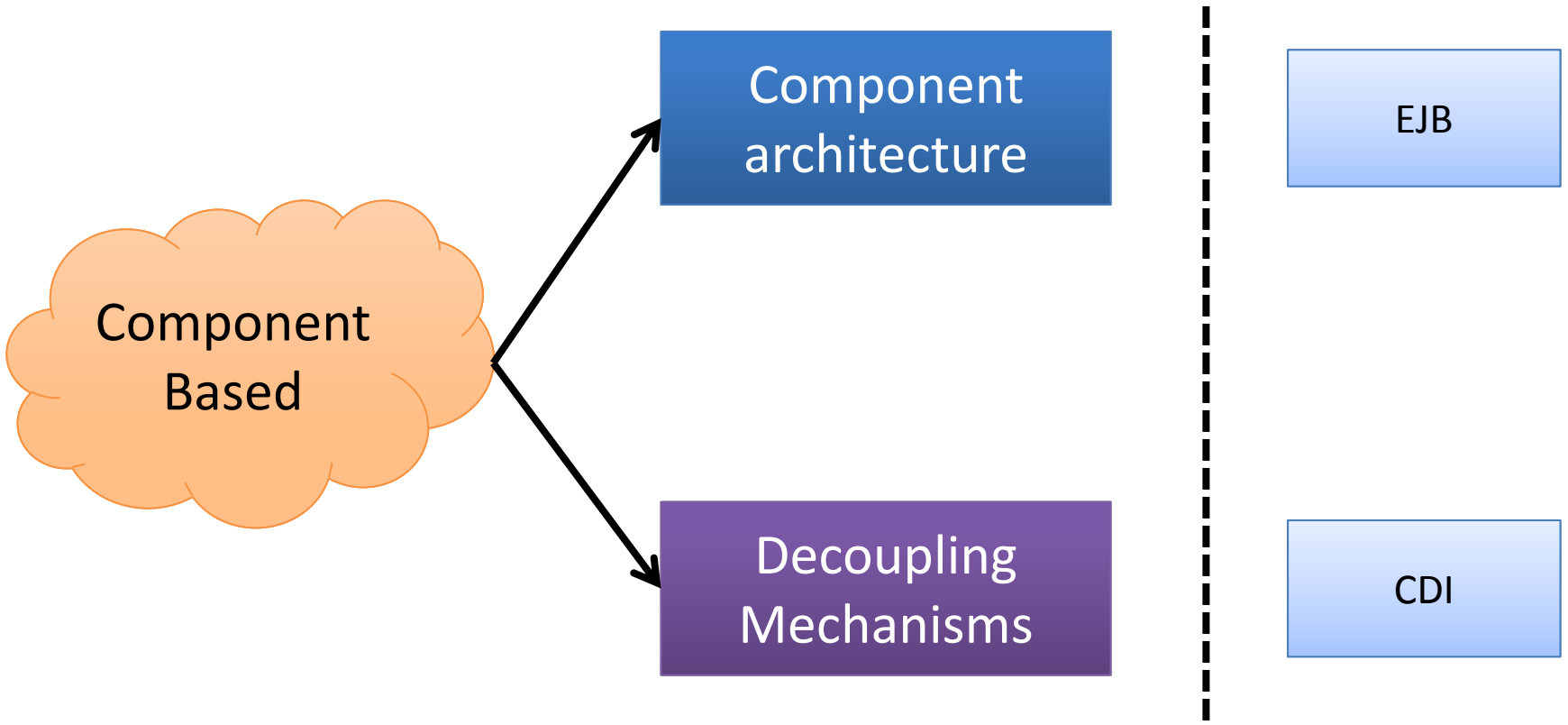
Java EE Overview

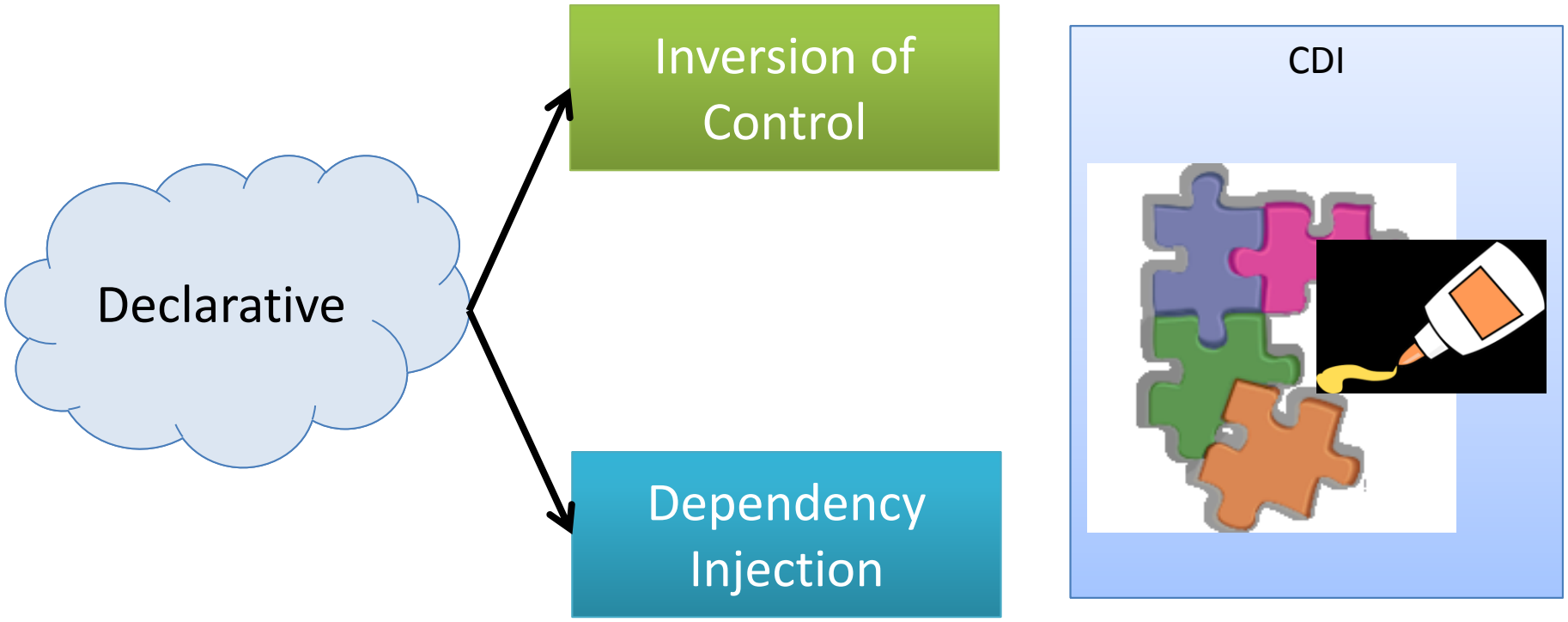



Web
HTML
Abstraction
HTTP
Abstraction
JSP
JSF
Servlet

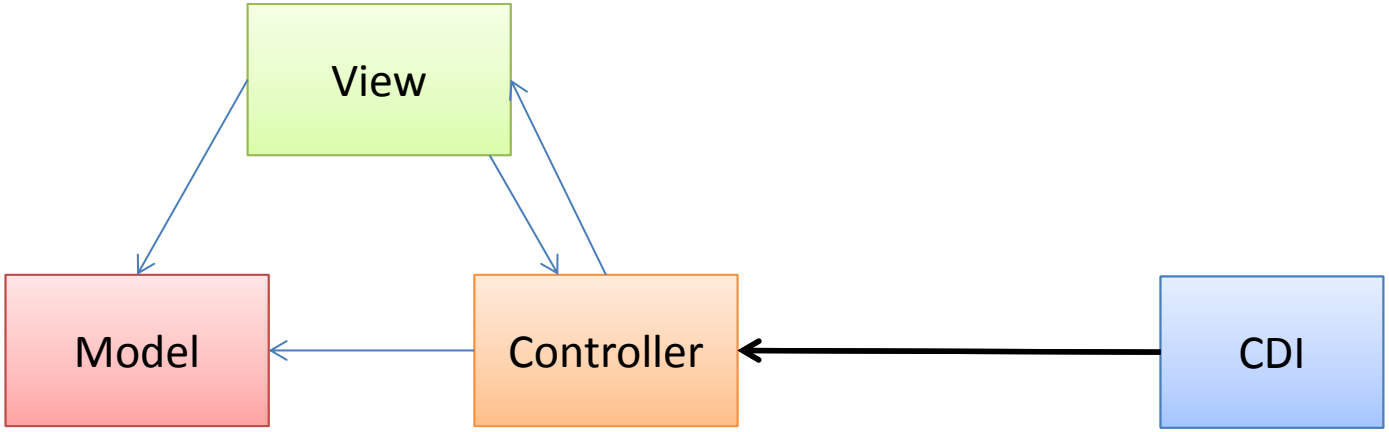

MVC
View
Model
Controller
Other
JSP
JSF
Servlet
EJB
CDI
Bean Validation
JPA


Component Based
Component 1
Component 3
Component 2
Packaging & Deployment
JEE Container
«ExecutionEnvironment»
Component 1
Component 2
HTTP,
REST,
Etc.
JEE Container
«ExecutionEnvironment»
Component 3


Component Based
Component architecture
Decoupling Mechanisms
EJB
CDI


Declarative
Inversion of Control
Dependency Injection
CDI

Context and Dependency Injection (CDI)



https://netbeans.org/kb/docs/javaee/cdi-inject.html
http://docs.oracle.com/javaee/6/tutorial/doc/gjbnr.html

“Classic” programming

- The programmer has the control
  - The program calls the functions

```
BufferedReader reader = new BufferedReader(
                        new InputStreamReader(System.in));
String name = reader.readLine();
String lastName = reader.readLine();
Database.save(name, lastName);
```

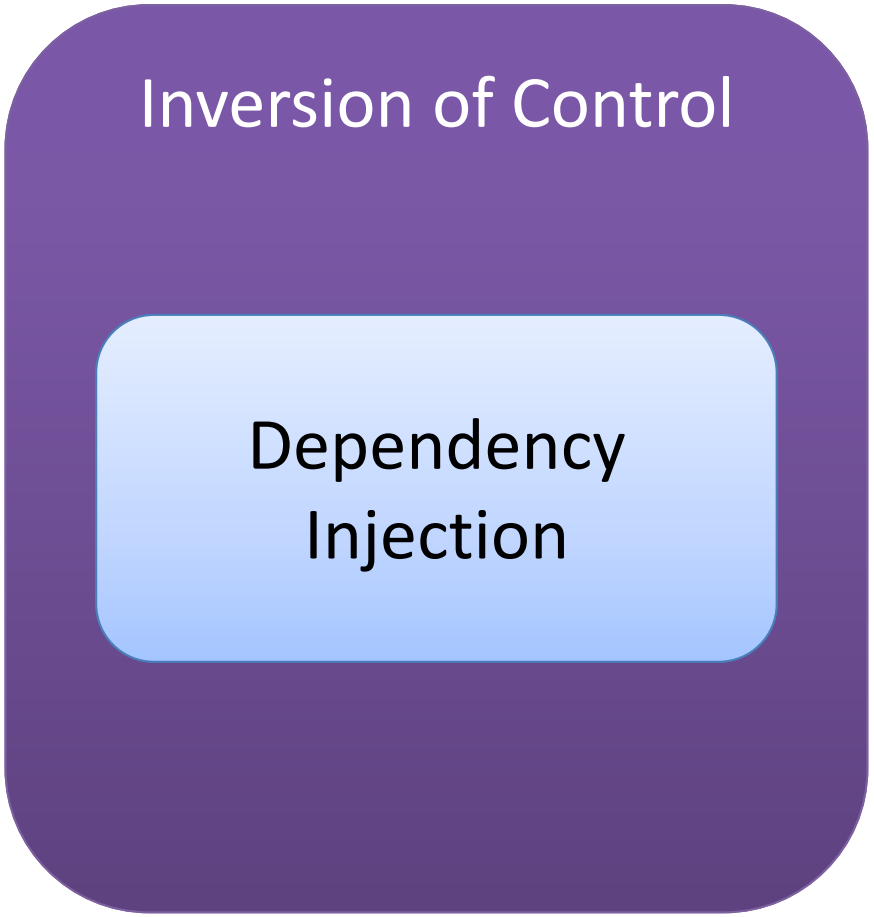
Inversion of Control (IoC)

- The framework has the control
  - The programmer implements methods
  - The framework calls them as needed

```
JFrame f = new JFrame();
JPanel p = new JPanel();
JTextField tName = new JTextField(20);
p.add(tName);
JTextField tLastName = new JTextField(20);
p.add(tLastName);
Button b = new Button("save");
b.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        Database.save(tName.getText(), tLastName.getText());
    }
});
p.add(b);
f.add(p);
f.setSize(400,300);
f.setVisible(true);
```

http://martinfowler.com/bliki/InversionOfControl.html

Dependency Injection (DI)



“Classic” Programming (no DI)

```java
interface Persistence {
    void save(Object obj);
}

class XMLPersistence implements Persistence {
    @Override
    public void save(Object obj) {
        // Saves obj in XML format
    }
}

class SQLPersistence implements Persistence {
    @Override
    public void save(Object obj) {
        // Saves obj in a SQL database
    }
}
```

```java
class DataProcessor {
    XMLPersistence p = new XMLPersistence();

    public void execute() {
        Data obj = new Data();
        // manipulate obj's data
        p.save(obj);
    }
}
```

```java
DataProcessor proc = new DataProcessor();
proc.execute();
```

Disadvantages?

Dependency Injection

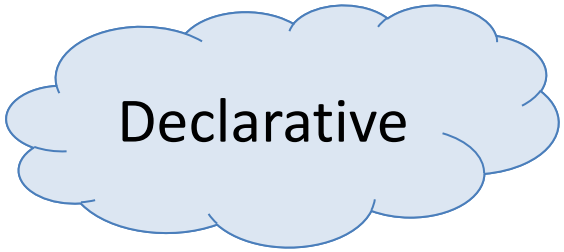
```java
interface Persistence {
    void save(Object obj);
}

class XMLPersistence implements Persistence {
    @Override
    public void save(Object obj) {
        // Saves obj in XML format
    }
}

class SQLPersistence implements Persistence {
    @Override
    public void save(Object obj) {
        // Saves obj in a SQL database
    }
}
```

```java
class DataProcessor {
    Persistence p;

    public DataProcessor(Persistence p) {
        this.p = p;
    }

    void execute() {
        Data obj = new Data();
        // manipulate obj's data
        p.save(obj);
    }
}
```

```java
DataProcessor proc = new DataProcessor(new XMLPersistence());
proc.execute();
```

Advantages? **Disadvantages?**

Context and Dependency Injection (CDI)



- Declarative way to inject dependencies
- Removes dependencies

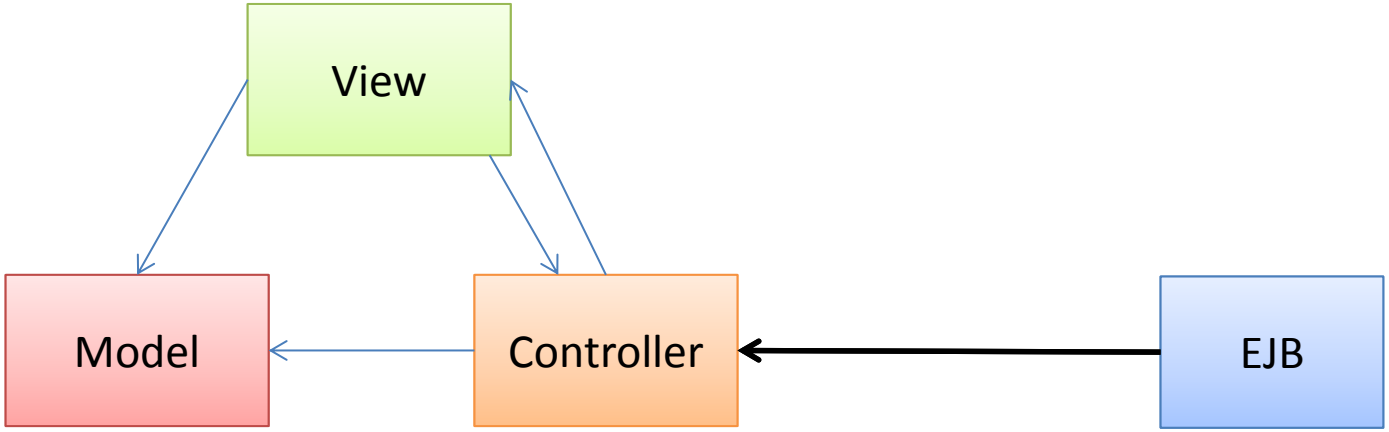
```
@Named
class DataProcessor {
    @Inject
    Persistence p;

    void execute() {
        Data obj = new Data();
        // manipulate obj's data
        p.save(obj);
    }
}
```

- DataProcessor is automatically instantiated by the Java EE server

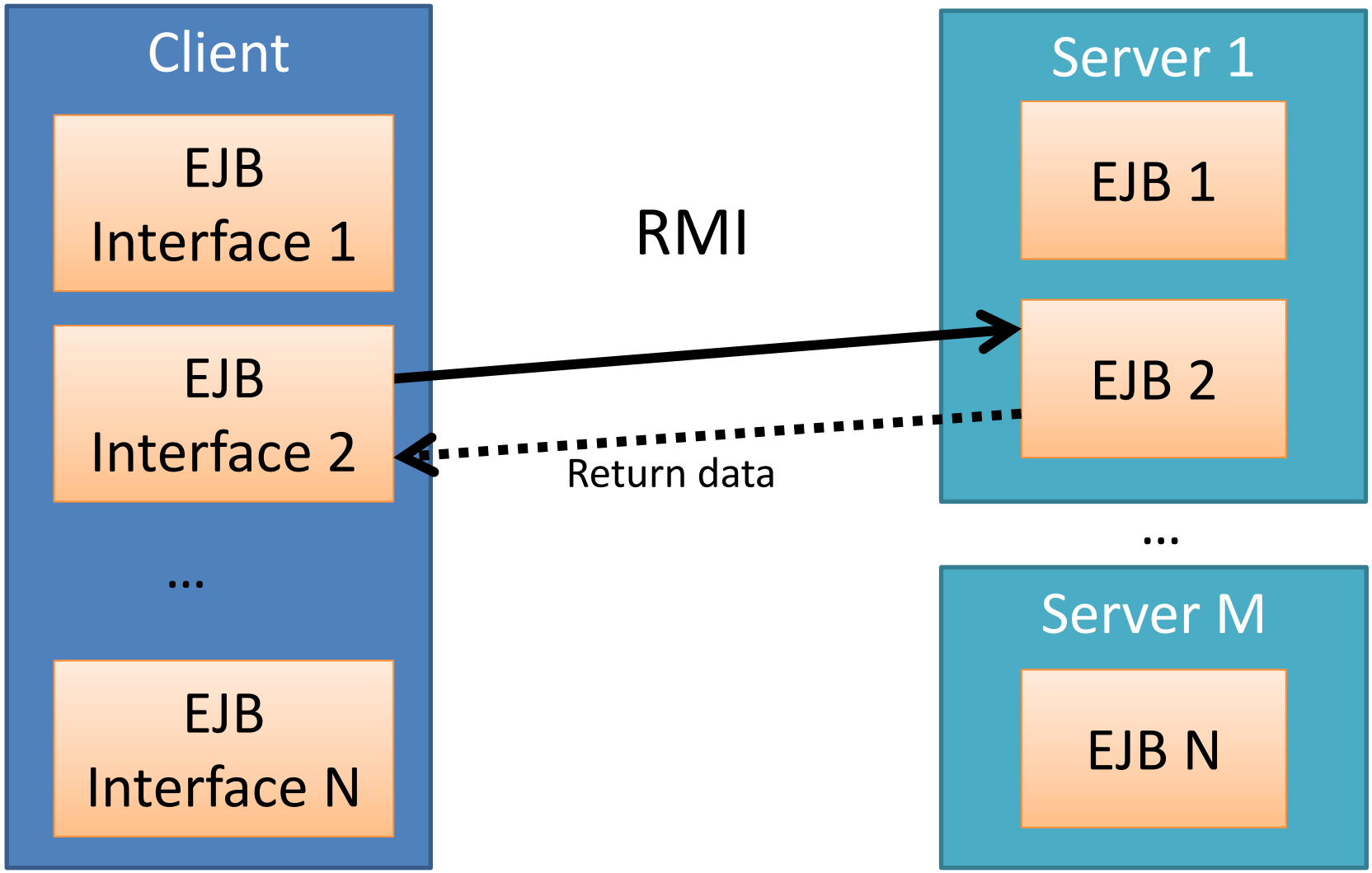
CDI Basic Annotations

`@Named("instanceName")`	Gives a name to a bean. The bean can be accessed with that name
`@Inject`	Automatically creates and assigns a bean to an attribute or parameter
`@RequestScoped`	The bean is accessible only during a single HTTP request
`@SessionScoped`	The bean is accessible during a session (multiple HTTP requests)
`@ApplicationScoped`	The bean is "global". It can be accessed by all users of an application
`@Dependent`	Default scope. Same life-cycle as the bean where is injected
`@ConversationScoped`	See http://docs.oracle.com/javaee/6/tutorial/doc/gjbbk.html

Enterprise Java Beans (EJB)



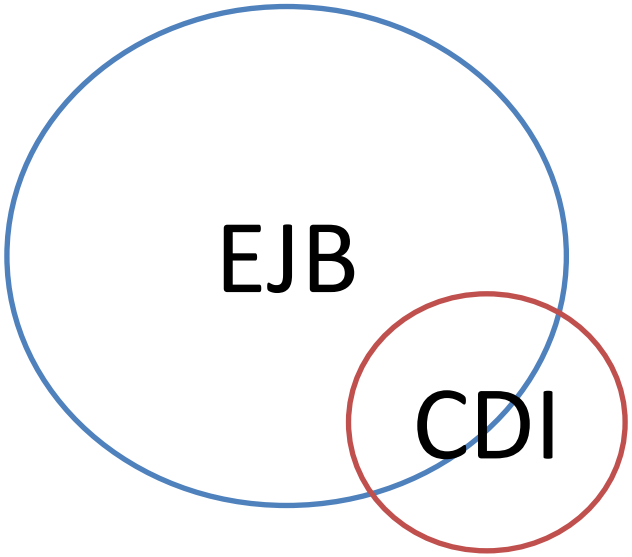
Enterprise Java Beans (EJB)



https://docs.oracle.com/javaee/7/tutorial/partentbeans.htm

EJB vs CDI

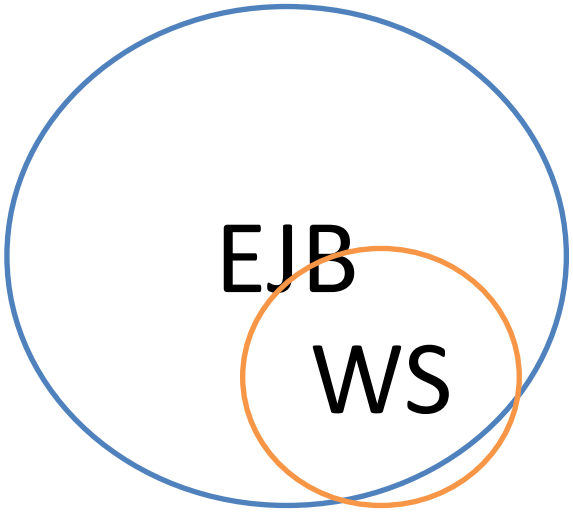
- CDI strengths: injection, lightweight
- EJB strengths: distributed objects



http://stackoverflow.com/questions/4684112/how-do-cdi-and-ejb-compare-interact
http://blog.dblevins.com/2012/11/cdi-when-to-break-out-ejbs.html

EJB vs Web Services (WS)

- EJB strengths: performance
- WS strengths: compatibility

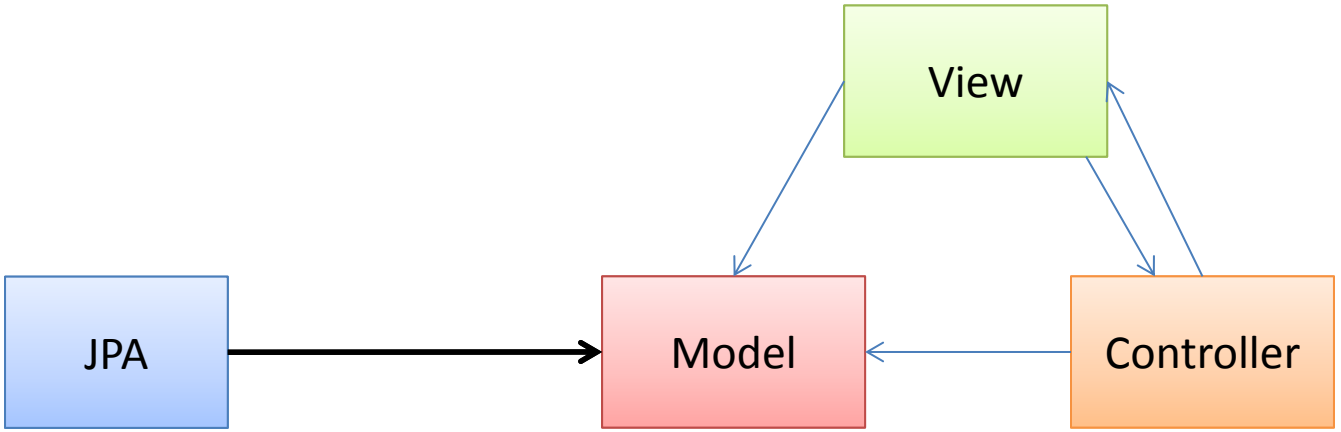


https://docs.oracle.com/javaee/7/tutorial/jaxws.htm
http://stackoverflow.com/questions/10198286/is-still-useful-to-implement-ejb-with-rmi-when-you-can-implement-web-services-s

EJB Basic Annotations

`@Stateless`	Session bean that does not keep a state
`@Stateful`	Session bean that does keeps a state
`@MessageDriven`	A bean that can process messages asynchronously
`@Entity`	Persistent bean

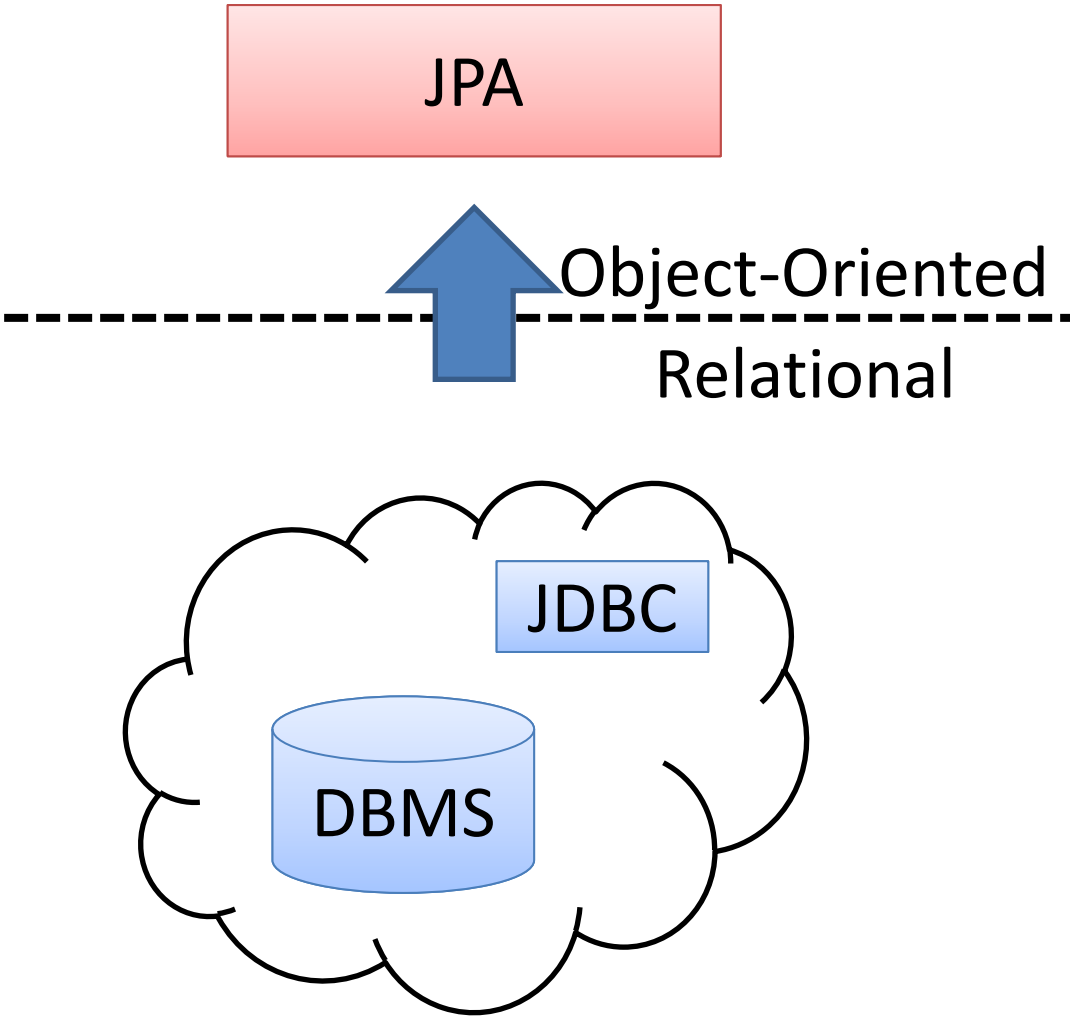
Java Persistence API (JPA)



http://docs.oracle.com/javaee/6/tutorial/doc/bnbpy.html

Persistence

- Database abstraction
- JDBC Abstraction
- Object-Relational Mapping



Main JPA Components

- Entities
  - IDs
  - Attributes
- Relations
  - 1..1 1..* *..1 *..*
  - Cascade operation
- Entity Manager
- Queries

Entities

Declares a class as a persistent entity

Recommended, but not mandatory

```
@Entity
public class Person implements Serializable {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    private String name;

    // getters and setters, hashCode, equals, toString
```

Declares an attribute as the entity's ID

Persisted by default

Defines how to generate the ID

- http://www.objectdb.com/java/jpa/entity/generated

Relations

```java
@Entity
public class Company implements Serializable {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;

    @OneToMany(mappedBy = "company")
    private List<Division> divisions;
    // Getters + setters + other methods
```

```java
@Entity
public class Division implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;
    private String TaxID;

    @ManyToOne
    private Company company;
    // Getters + setters + other methods
```

- @OneToOne
- @OneToMany
- @ManyToOne
- @ManyToMany

- http://www.javaworld.com/article/2077819/java-se/understanding-jpa-part-2-relationships-the-jpa-way.html?page=2
- http://stackoverflow.com/questions/14130041/jpa-persist-entities-with-one-to-many-relation

Validation

- Annotations that constrain attribute values

```
@Entity
public class Division implements Serializable {

    @NotNull
    private String TaxID;
```

https://docs.oracle.com/javaee/7/tutorial/bean-validation001.htm

Spring Framework

Java EE vs. Spring Framework Features/APIs

https://www.slideshare.net/kelapure/2012-0406v2tdp1163java-eevsspringshootoutfinal

Ejemplo Spring

git clone https://bitbucket.org/jpavlich/pw.git

Ejercicio

- Usar proyecto base

- Crear todas las entidades del proyecto semestral.