A Component Based Approach

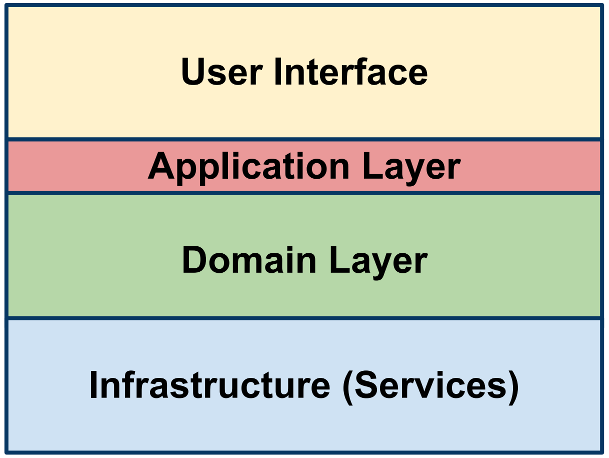
JSF Slides #5

Characterization Review

Much more of standard (non-web) OO-programming. Well known concepts of objects, components and listeners

High abstraction level, possible a bit lack of control Normally not accessing HTTP request etc. (but pops-up...)

Application Layers vs JEE Stack



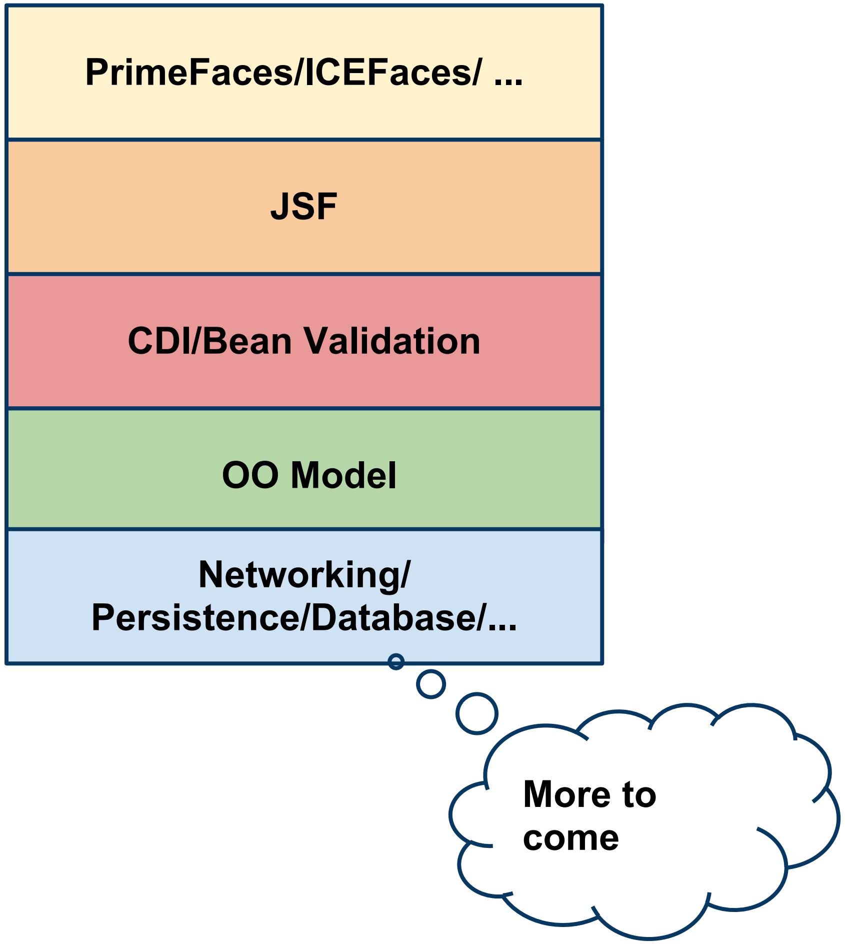
User Interface

Application Layer

Domain Layer

Infrastructure (Services)

Domain driven application layering



PrimeFaces/ICEFaces/ ...

JSF

CDI/Bean Validation

OO Model

Networking/ Persistence/Database/...

More to come

Design and MVC

JSF/CDI/Bean Validation is not a complete framework, no default MVC design...

Full Frameworks
- Spring
- Seam
...

For now we have to design ourselves, so yet another in house MVC- solution

Managed Bean Roles in MVC

Model-bean (session scope) represents data (doubtful?)
- Model objects representing collections seems ok

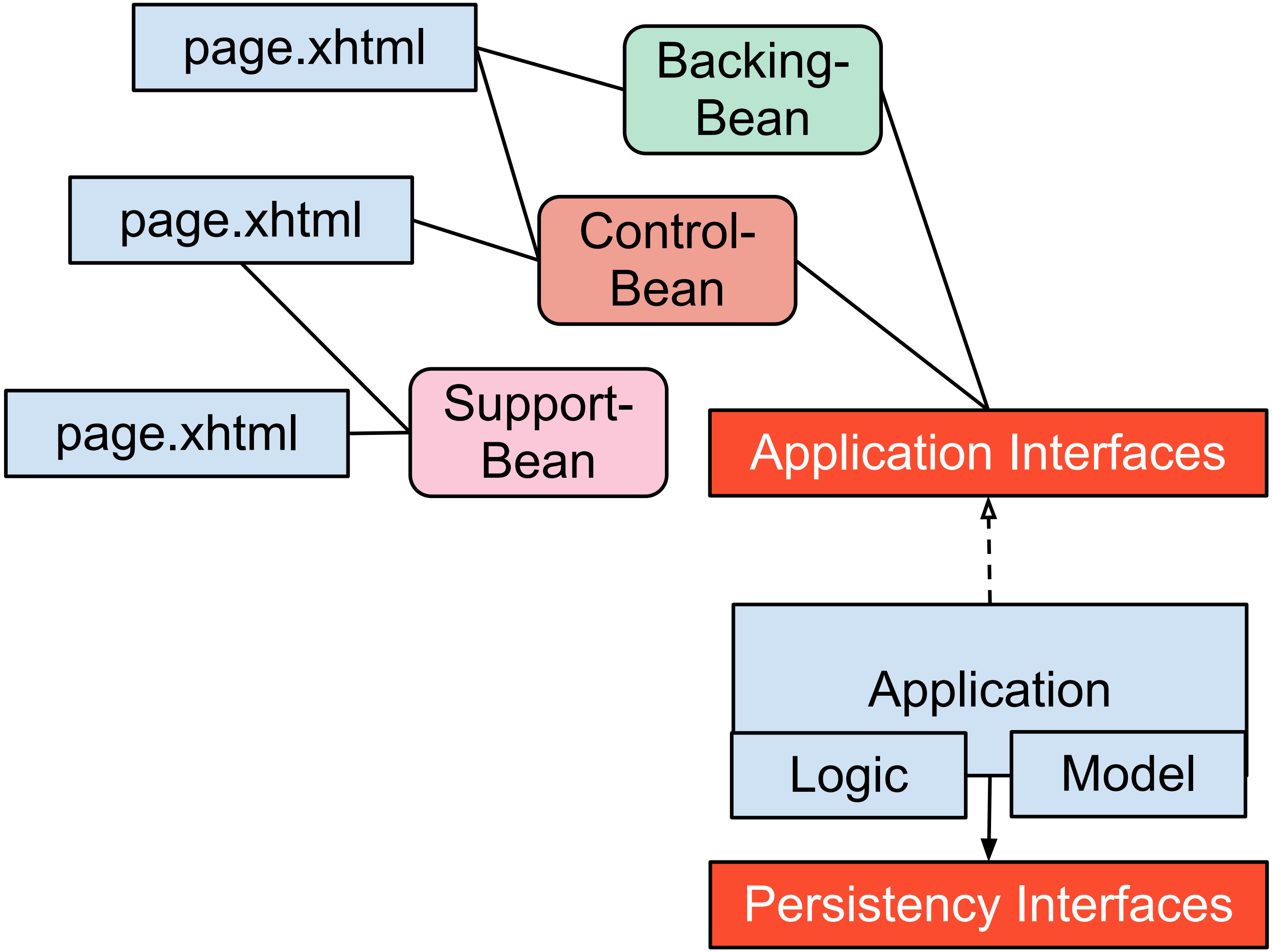
Backing (page)-bean (request) Support view. 1:1 mapping to page. Possible have listener methods.

Controller-bean (request). Part of control. Execute application or business logic. Return navigation outcome

Support-bean (session, application) Support many pages (populate drop-down lists etc.)

Utility-bean (application) General functionallity used by many pages/applications (file upload)

JSF Application Design



page.xhtml

page.xhtml

page.xhtml

Backing-Bean

Control-Bean

Support-Bean

Application Interfaces

Application

Logic

Model

Persistency Interfaces

Authorization

Have seen usage of filters (very simple) and OAuth (complex)..

Now for the predefined JEE authorization technique, using realms

A **realm** is a security policy domain defined for a web or application server. A realm contains a collection of users, who may or may not be assigned to a group

Types of realms (supported by GlassFish and Tomcat)
- **file**, Stores user information in a file. This is the default realm when you first install the GlassFish Server
- **ldap**, Stores user information in an LDAP directory
- **jdbc**, Stores user information in a database
...

Realms, Users, Groups and Roles

For now we use the file realm (database advanced sample later)

Steps

-Create users and groups in GlassFish file realm (using Admin console) .. manual for now
-Create roles in application (defined in glassfish-web.xml)
-Map roles to users and groups (also glassfish-web.xml)

Authorization in Application

```
// In some backing bean connected to login page
// Using default mechanism and HTTPServletRequest (request)
…
if (request.getUserPrincipal() != null) {
  return "login-success";
}
request.login(username, password);

if (request.isUserInRole("productManager")){
   user = new User(username);
   Logger.getAnonymousLogger().log(Level.INFO, "Successfully logged
in
                                              {0}", username);
   return "login-success";
} else {
   return "login-fail";
}
```

Logged in as …

Simple approach in pages

```
<h:outputText value="Logged in as #{request.remoteUser}" />
```