95-702 Distributed Systems

Components

Components

- Components represent a natural evolution from distributed object computing.
- First, we will review problems with distributed objects (Java RMI and CORBA).
- Along the way we will examine the requirements that led to component based approaches.

Issues with Object Oriented Middleware

- Implicit dependencies are not clear
  - The provided interfaces are a contract between the client and the server but say nothing about the requirements that the service may have.
  - E.g. Does it use a registry? Does it require other services such as transactions, security, persistence, etc.?

Issues with Object Oriented Middleware

- Since implicit dependencies are not specified it is difficult to ensure the safe composition of a configuration, to replace an object with another, and for third party developers to implement one particular element in a distributed configuration.
- Requirement: Specify not only the interfaces offered by an object but also the dependencies that object has on other objects in the distributed configuration.

Issues with Object Oriented Middleware

- There is a lack of separation of concerns.
  - Programmers writing distributed objects are faced with dealing with non-functional concerns – security, transactions, coordination and replication, etc.
  - This tangling of concerns increases the complexity of distributed system programming.

Issues with Object Oriented Middleware

- Requirement: The complexities of dealing with distributed systems services should be hidden wherever possible from the programmer.

Issues with Object Oriented Middleware

- There is no support for deployment.
  - Objects must be deployed manually on individual machines.
  - This is a tiresome and error prone process in large heterogeneous environments.
- Requirement: Middleware platforms should provide support for deployment with the complexities hidden from the programmer.

Three Requirements

(1) Specify not only the interfaces offered by an object but also the dependencies that object has on other objects in the distributed configuration.

(2) The complexities of dealing with distributed systems services should be hidden wherever possible from the programmer.

(3) Middleware platforms should provide support for deployment with the complexities hidden from the programmer.

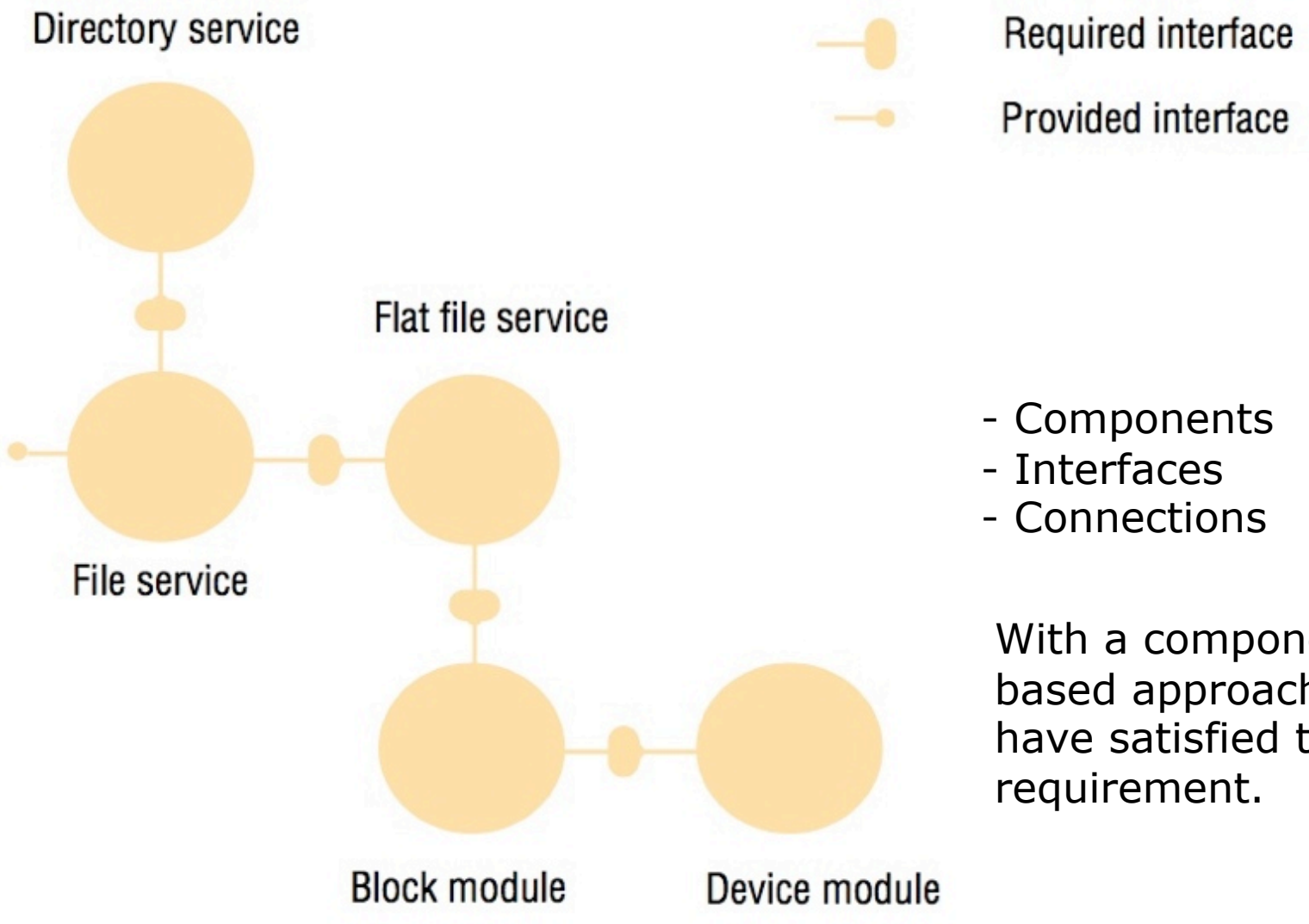
Definition of Component

- Component: A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. From Szyperski's book on component sofware.
- A given component specifies both its interfaces provided to the outside world and its dependencies on the other components in the distributed environment.

Software Architecture

- In a given component configuration, every required interface must be bound to a provided interface of another component. This is also referred to as a *software architecture*.

Software Architecture Example



- Components
- Interfaces
- Connections

With a component based approach we have satisfied the first requirement.

Separation of Concerns!!

- We need to remove complexity and simplify the development and deployment of distributed applications.
- In the distributed systems community we have seen the emergence of Enterprise Java Beans, CORBA Component Model (CCM) and Service Component Architecture (SCA) for Service Oriented Architectures

SOC Provided by Containers

- The task of a container is to provided a managed server-side hosting environment for components and to provide the necessary separation of concerns where components deal with application concerns and the container deals with the distributed systems and middleware issues.

Application Server

- Middleware that supports the container pattern and the separation of concerns implied by this pattern is known as an *application server*.

Application servers

Technology	*Developed by*	*Further details*
WebSphere Application Server	IBM	[www.ibm.com]
Enterprise JavaBeans	SUN	[java.sun.com XII]
Spring Framework	SpringSource (a division of VMware)	[www.springsource.org]
JBoss	JBoss Community	[www.jboss.org]
CORBA Component Model	OMG	[Wang *et al.* 2001]
JOnAS	OW2 Consortium	[jonas.ow2.org]
GlassFish	SUN	[glassfish.dev.java.net]

Next up

- Enterprise Java Beans (EJB's) is a specification of a server-side managed component architecture and a major part of the Java Platform, Enterprise Edition (Java EE).