INTRODUCTION TO SOFTWARE ENGINEERING

LECTURE – 25

MAY 22, 2017

TOPICS COVERED

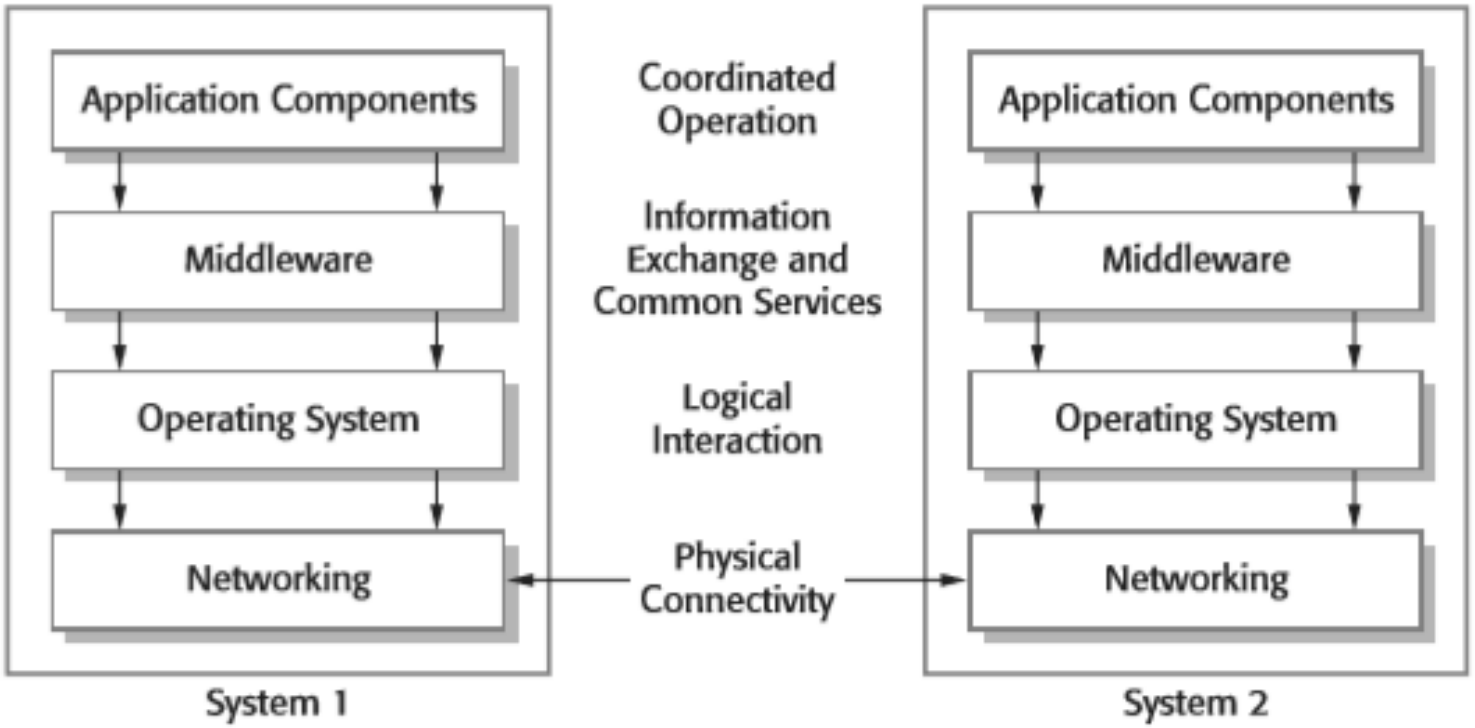
- **Distributed Software Engineering**
- **Reuse Based Software Engineering**
- **Configuration Management**

Distributed Software Engineering

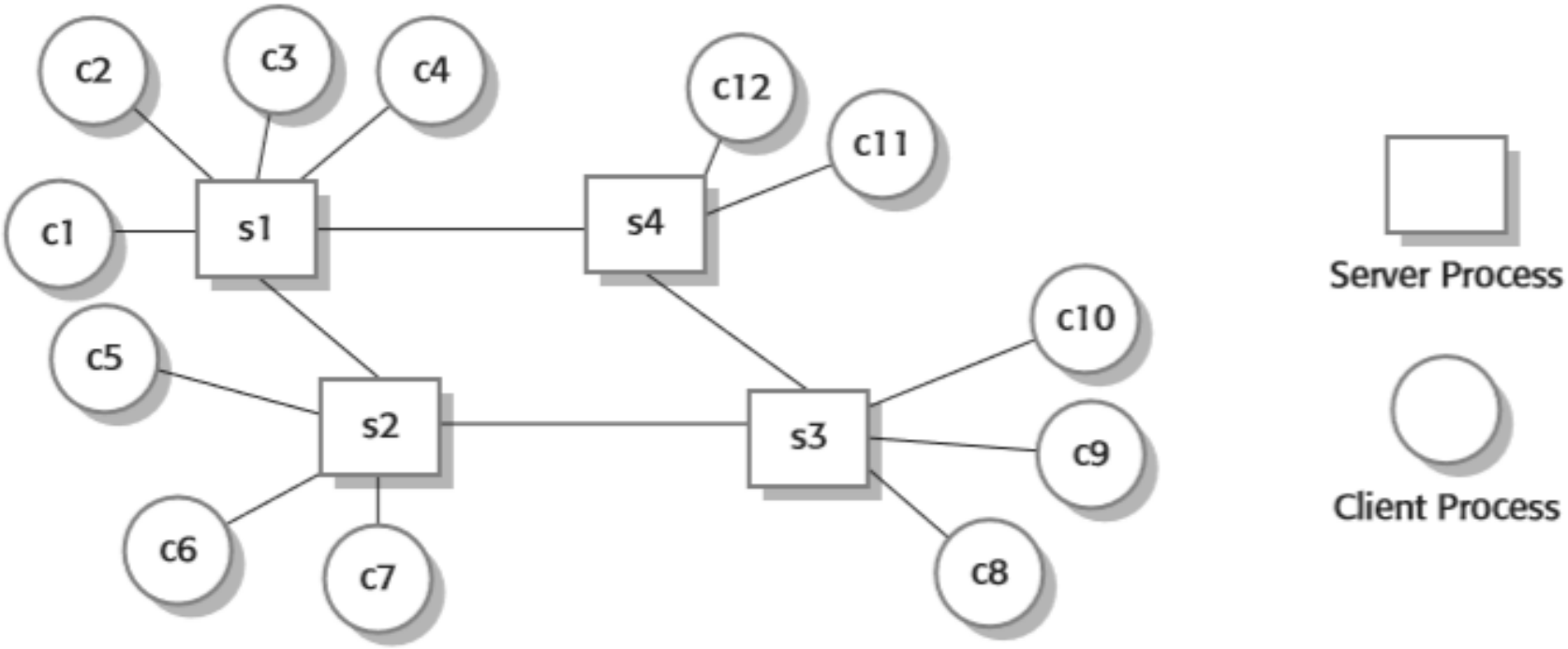
DISTRIBUTED SYSTEMS ISSUES

- **Transparency**
- **Openness**
- **Scalability**
- **Security**
- **Quality of Service**
- **Failure Management**

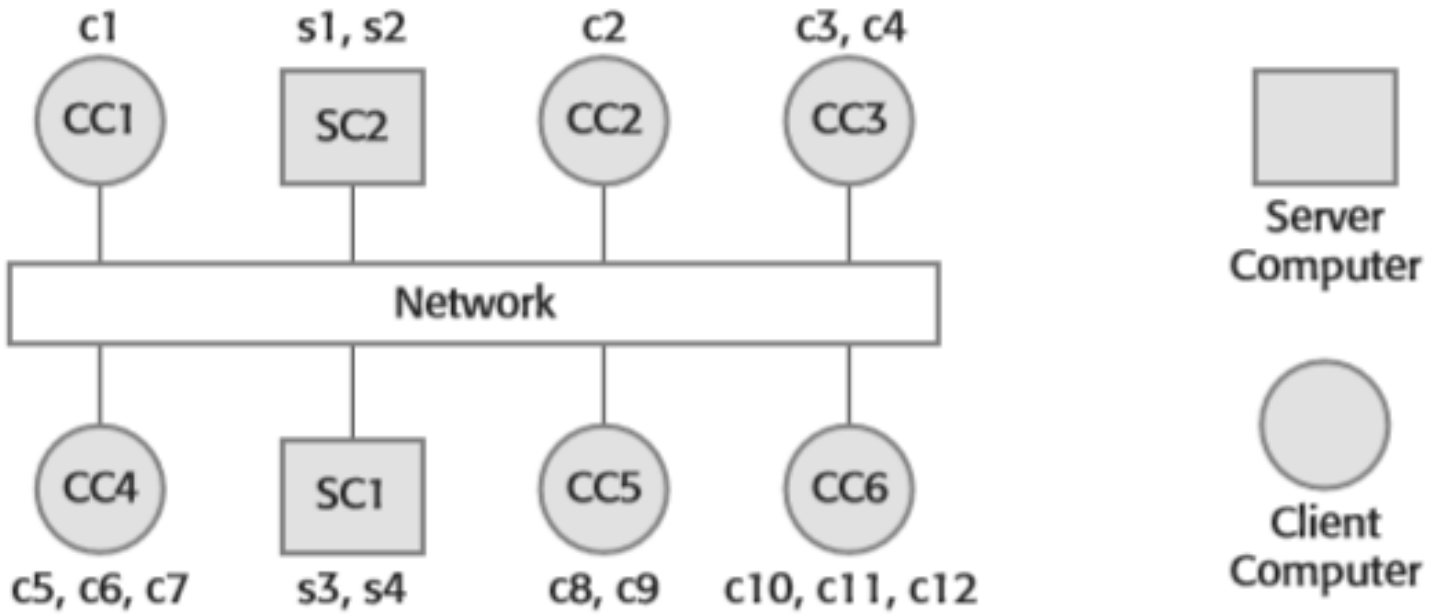
MIDDLEWARE


Application Components
Middleware
Operating System
Networking
System 1
Coordinated Operation
Information Exchange and Common Services
Logical Interaction
Physical Connectivity
Application Components
Middleware
Operating System
Networking
System 2

CLIENT-SERVER INTERACTION



CLIENT-SERVER COMPUTING



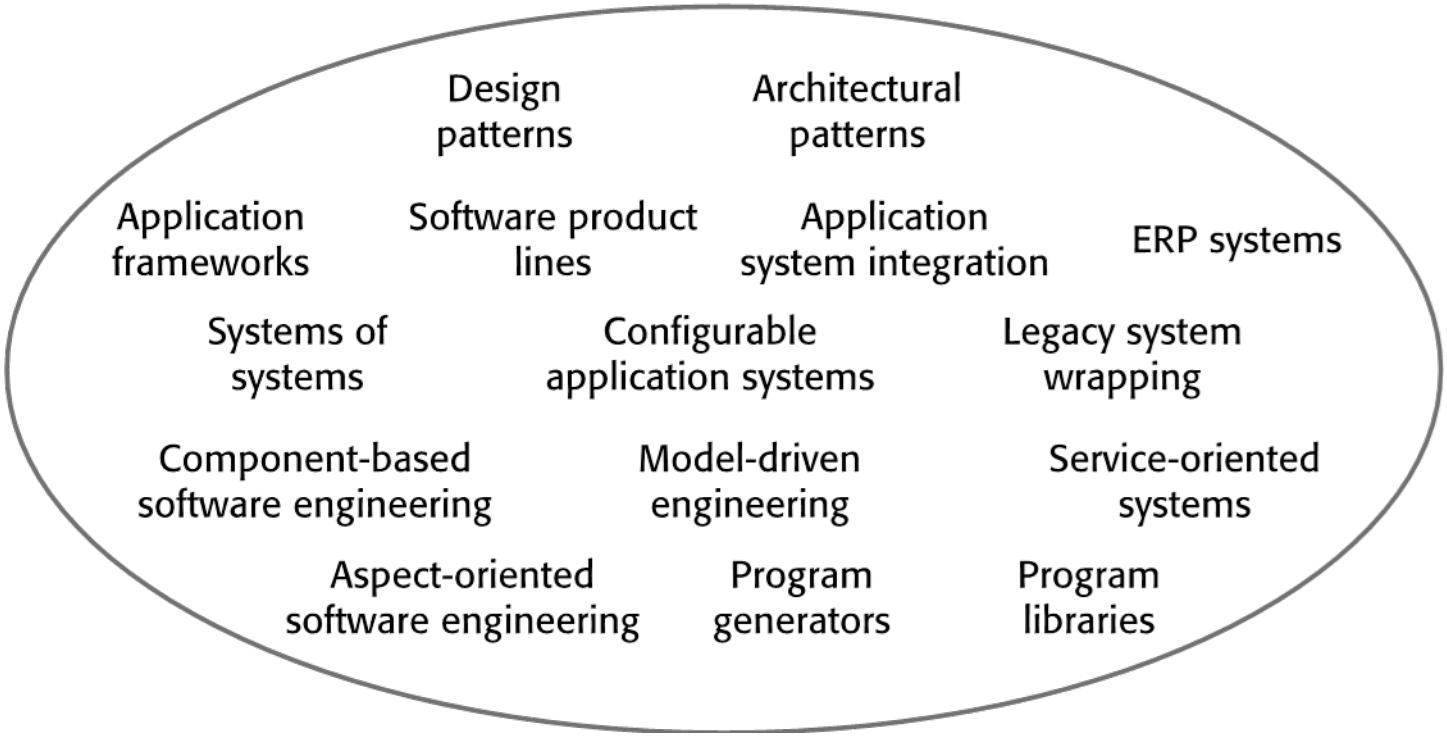
REUSE-BASED SOFTWARE ENGINEERING

- **System reuse**
  - Complete systems, which may include several application programs may be reused.
- **Application reuse**
  - An application may be reused either by incorporating it without change into other or by developing application families.
- **Component reuse**
  - Components of an application from sub-systems to single objects may be reused.
- **Object and function reuse**
  - Small-scale software components that implement a single well-defined object or function may be reused.

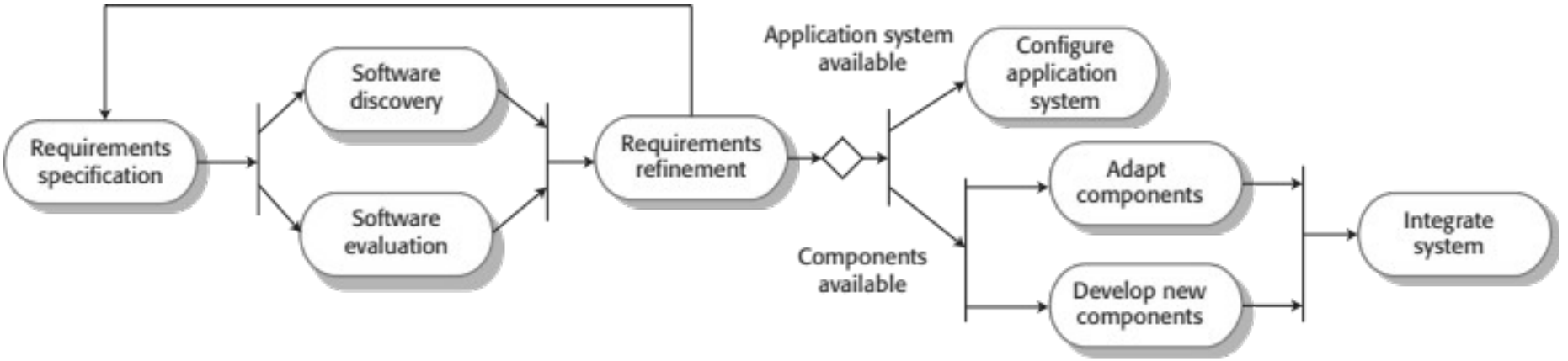
THE RE-USE LANDSCAPE

- **Although reuse is often simply thought of as the reuse of system components, there are many different approaches to reuse that may be used**
- **Reuse is possible at a range of levels from simple functions to complete application systems**
- **The reuse landscape covers the range of possible reuse techniques**

THE RE-USE LANDSCAPE



REUSE-ORIENTED SOFTWARE ENGINEERING



KEY PROCESS STAGES FOR ACQUISITION

- **Requirements specification**
- **Software discovery and evaluation**
- **Requirements refinement**
- **Application system configuration**
- **Component adaptation and integration**

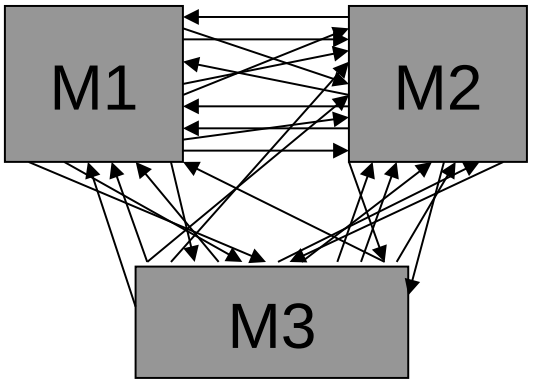
DOMAIN ENGINEERING FOR REUSE IN SOFTWARE ENGINEERING

Domain Engineering entails:

- **Domain Analysis**
  - Commonalities and differences of systems in a domain are discovered and recorded
- **Domain Implementation**
  - It means the use of information collected in domain analysis to create reusable components and new systems

CHARACTERISTICS OF GOOD DESIGN

- **Component independence**
  - High cohesion
  - Low coupling
- **Exception identification and handling**
- **Fault prevention and fault tolerance**
- **Design for change**

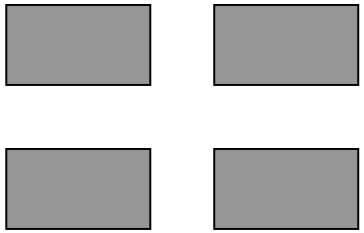


COHESION

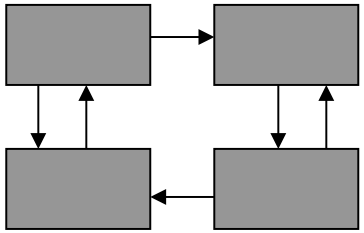
- **Definition**
  - The degree to which all elements of a component are directed towards a single task
  - The degree to which all elements directed towards a task are contained in a single component
  - The degree to which all responsibilities of a single class are related
- **Internal glue with which component is constructed**
- **All elements of component are directed toward and essential for performing the same task**

COUPLING

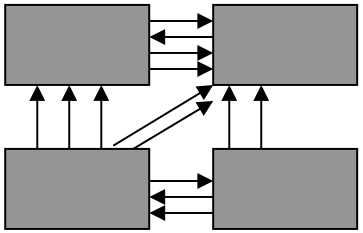
- **The degree of dependence such as the amount of interactions among components**



No dependencies



Loosely coupled
some dependencies



Highly coupled
many dependencies

Q&A