



ILLINOIS INSTITUTE
OF TECHNOLOGY

Transforming Lives. Inventing the Future.

www.iit.edu

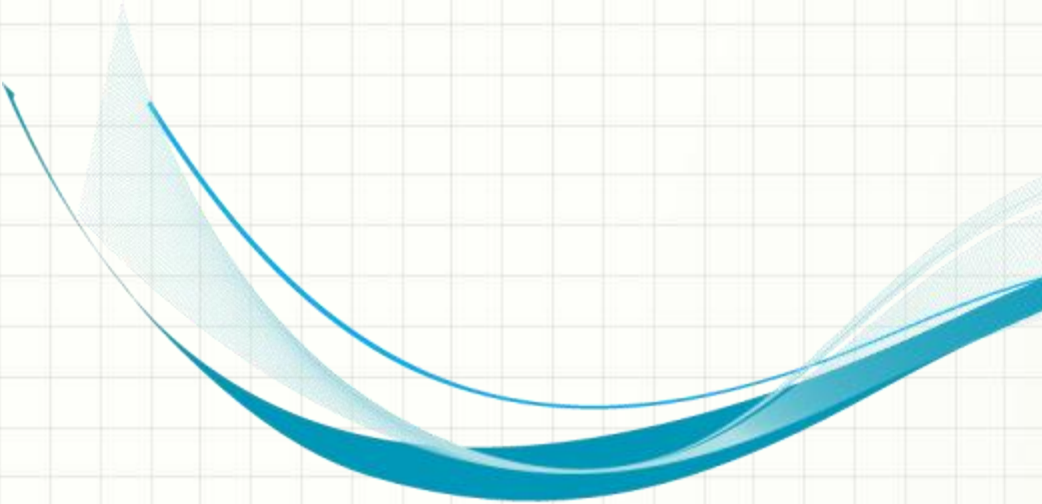
SOFTWARE ENGINEERING

CS 487

Prof. Dennis Hood
Computer Science

Lesson Overview

- Service-Oriented Architecture
- Reading
 - Ch. 17 – Distributed Software Engineering
 - Ch. 18 – Service-oriented Software Engineering
- Objectives
 - Systems are collections of cooperating sub-systems
 - As discussed previously in several contexts, the overall success of a system is dependent in large part on the predictable and reliable interaction with the various sub-systems
 - Service-oriented architectures extend to include sub-systems to which the “main” system binds at runtime



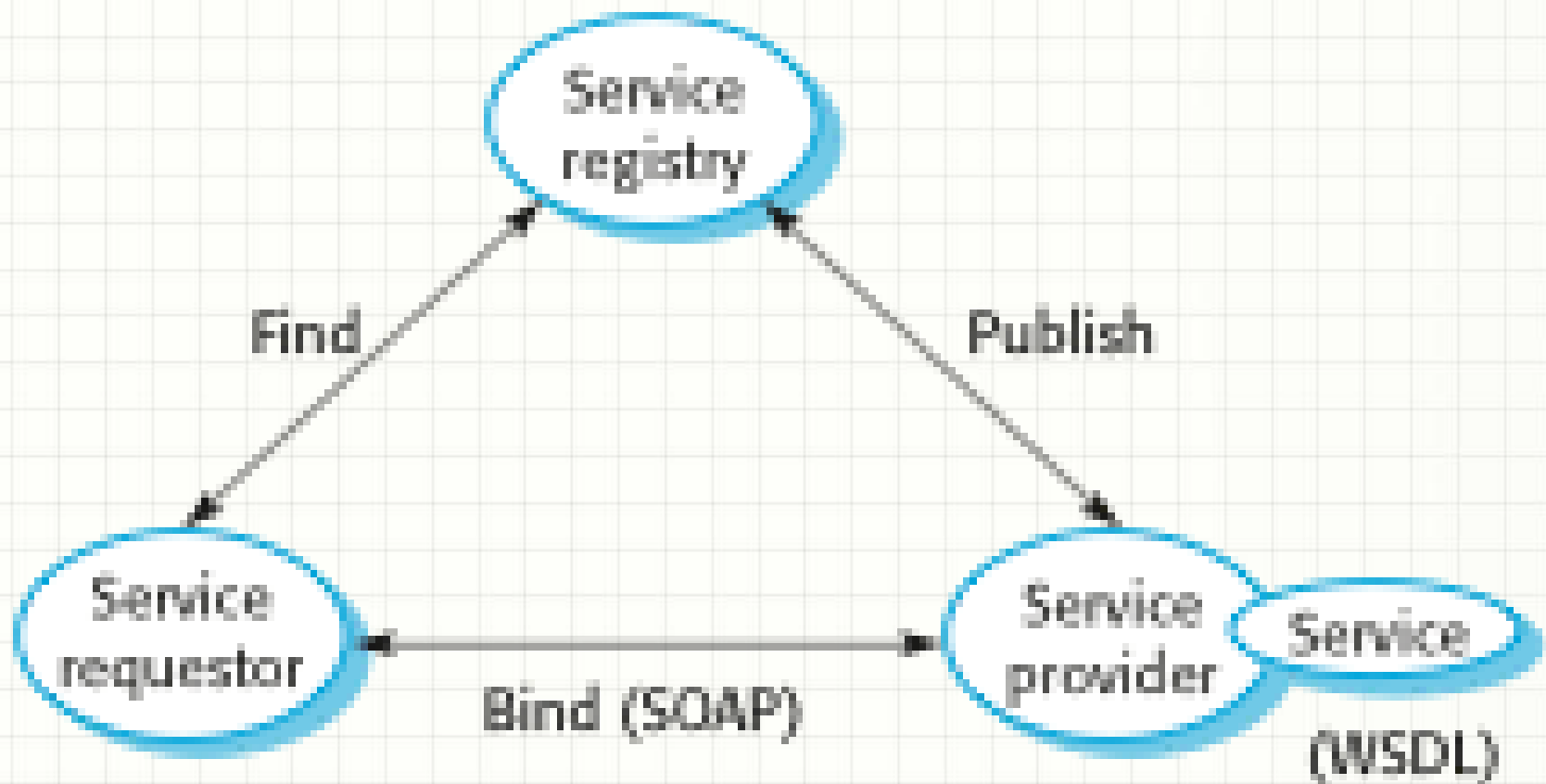
Week 12

Service-Oriented Architecture

Service-Oriented Architecture (SOA)

- Based on the notion of cooperating entities
 - Entities offer to perform services
 - Information can be made available in a controlled and managed way
- Defined interfaces must be published
- Allows for the creation of configurable, distributed, platform-independent systems

Service-Oriented Architecture



Web Services

- A web service is an instance of a more general notion of a service:

“an act or performance offered by one party to another. Although the process may be tied to a physical product, their performance is essentially intangible and does not normally result in ownership of any of the factors of production.”
- The provision of the service is independent of the application using the service
- Service providers can develop specialized services and offer these to a range of service users from different organizations

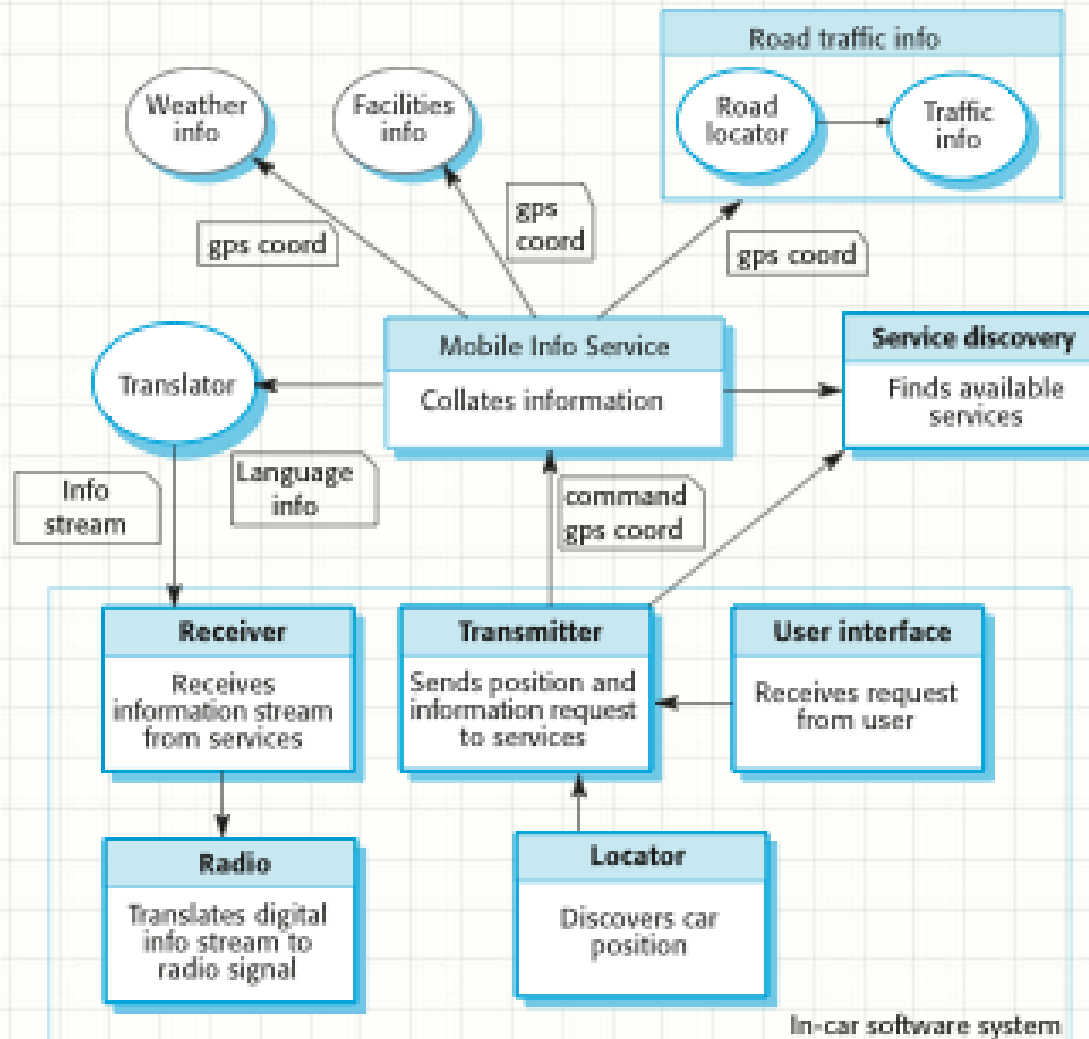
SOA-related Standards

- SOAP – SOA Protocol
 - Message interchange
 - Supports communication between services
- WSDL – Web Services Definition Language
 - Service interface definition
 - Establishes definition for service operations and bindings
- WS-BPEL – Web Services Business Process Execution Language
 - Workflow language

Benefits of SOA

- Services can be provided locally or outsourced to external providers
- Services are language-independent
- Investment in legacy systems can be preserved
- Inter-organizational computing is facilitated through simplified information exchange
- Reuse

Ex.: Car Information System



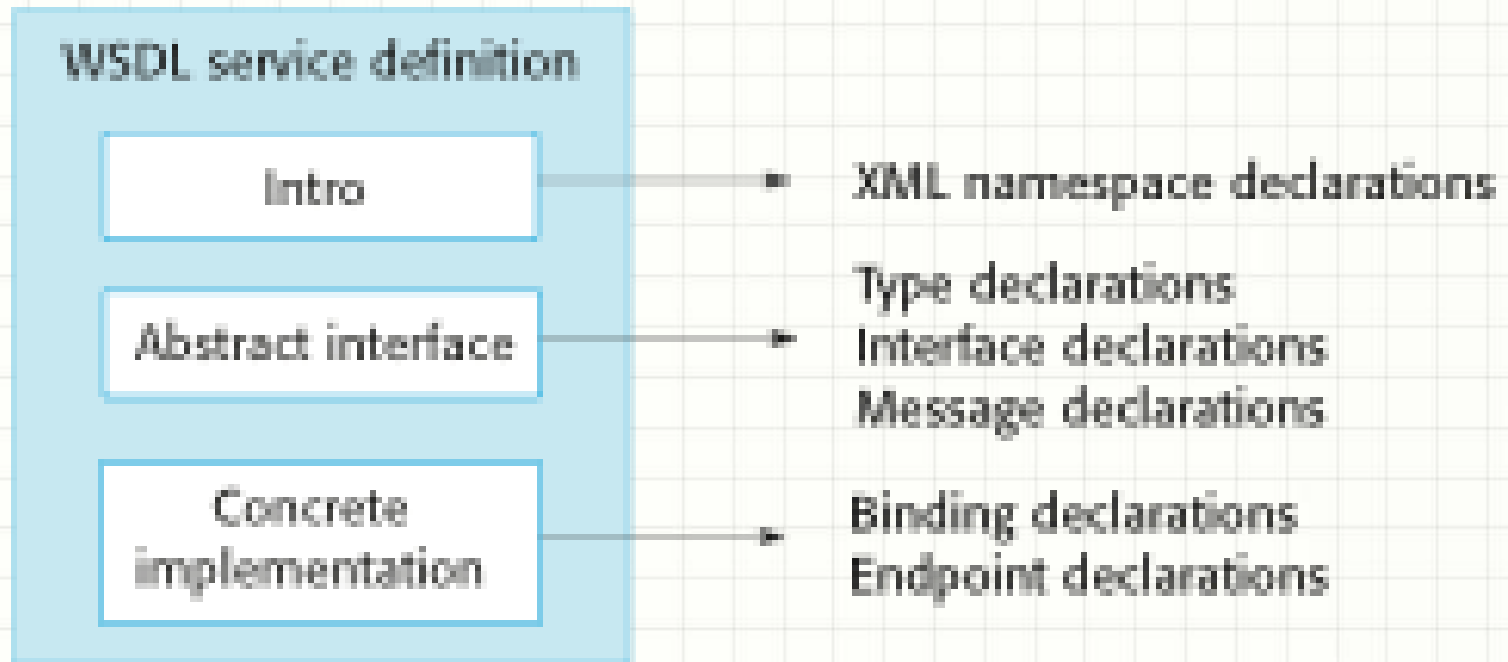
Ex.: Car Info Systems (cont.)

- It is not necessary to decide when the system is programmed or deployed what service provider should be used or what specific services should be accessed
 - As the car moves around, the in-car software uses the service discovery service to find the most appropriate information service and binds to that
 - Because of the use of a translation service, it can move across borders and therefore make local information available to people who don't speak the local language

Reusable Services

- Once established, a service can be used by any number of “systems”
 - Well-defined interface
 - Consistent, reliable performance
 - Independent and loosely coupled
- WSDL specification
 - What – the interface description
 - How – details of how to communicate
 - Where – the location of the implementation

WSDL Specification



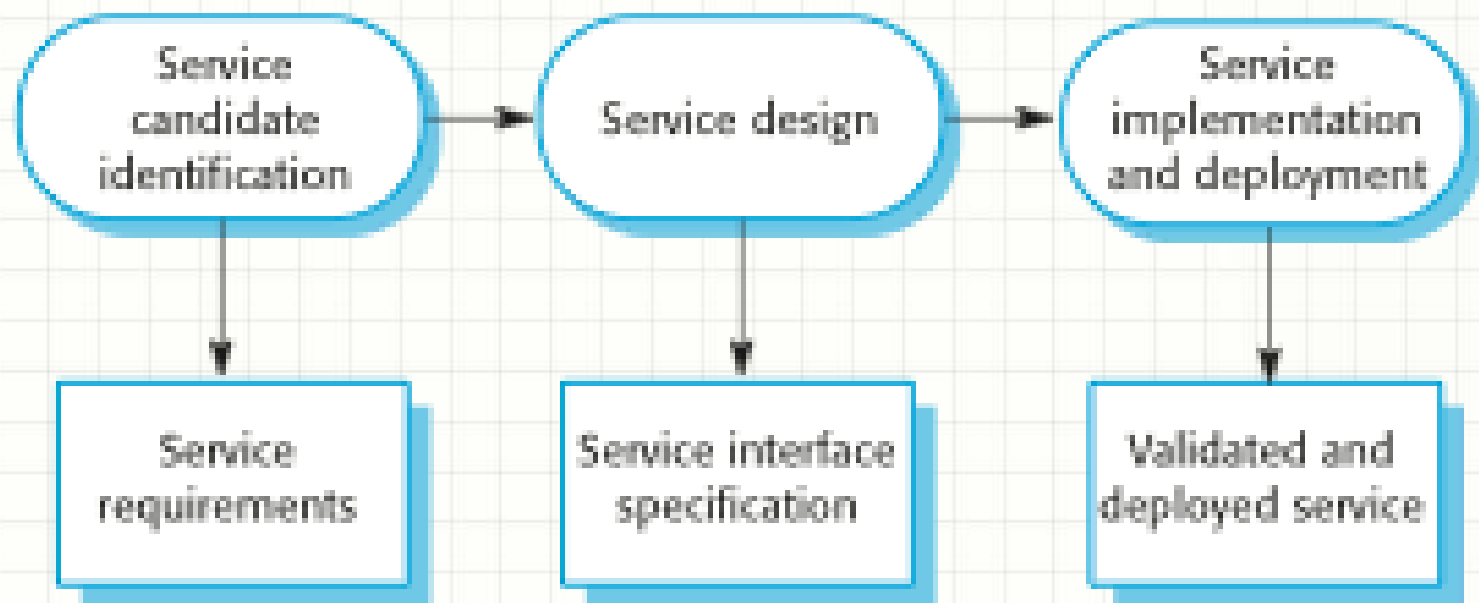
Service Types

- Utility services
 - General functionality
 - e.g., currency conversion
- Business services
 - Associated with a specific business function
 - e.g., student registration
- Coordination of process services
 - Support more general business processes
 - e.g., procurement process management

Service Engineering

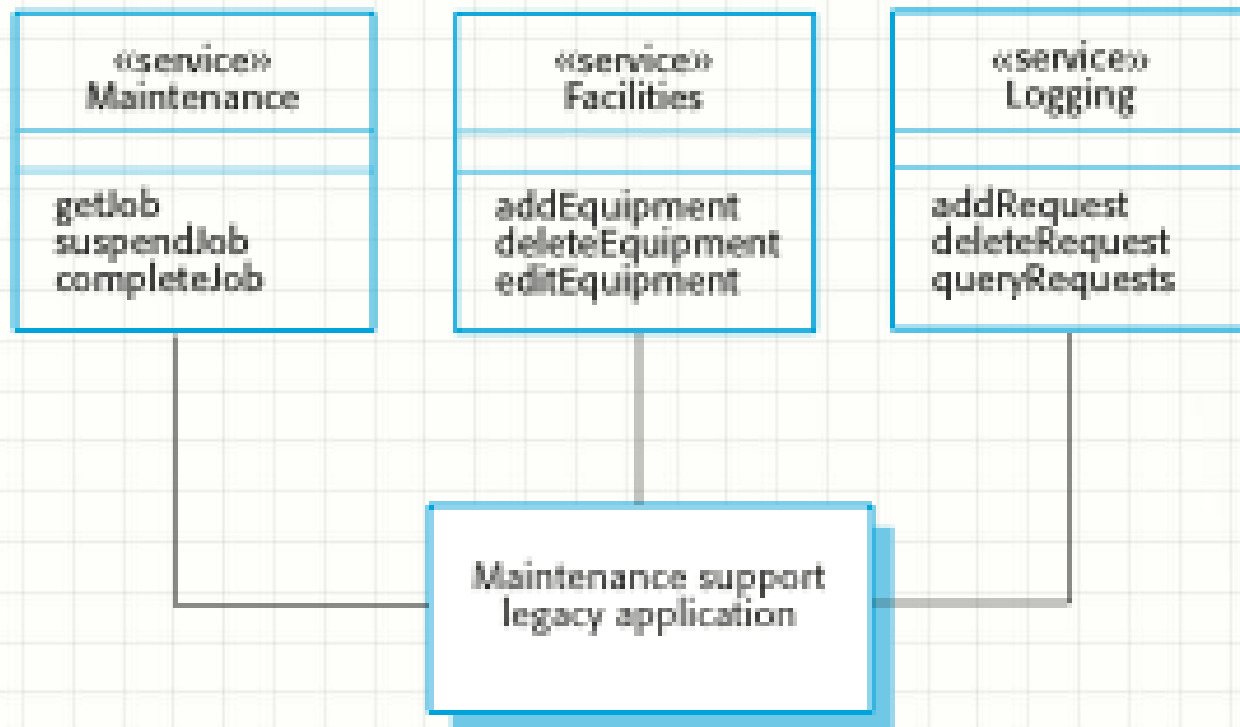
- The process of developing services for reuse in service-oriented applications
- The service has to be designed as a reusable abstraction that can be used in different systems
- Generally useful functionality associated with that abstraction must be designed and the service must be robust and reliable
- The service must be documented so that it can be discovered and understood by potential users

Service Engineering



Legacy System Wrappers

- Extend the life of legacy systems
 - Provide an interface in the form of a wrapper instead of rewriting (or “sun-setting”)



Service Testing Considerations

- Control or even access to external services is determined by the service provider
- An application may not always use the same service each time it is executed
- Performance may differ under varying loads – loads which can vary greatly depending on the number and frequency of service requests
- Testing of exception handling may depend on proper failure of dependent services