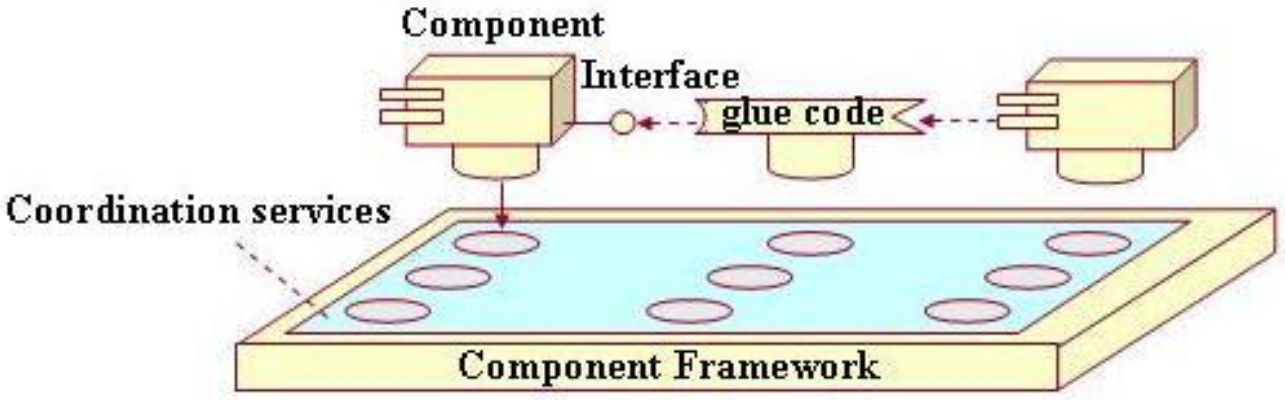
Tactics for Interoperability

主讲教师：王灿
Email: wcan@zju.edu.cn
TA: 李奇平 liqiping1991@gmail.com
Course FTP: ftp://sa:sa@10.214.51.13

Component-based Development

- A component is a self-contained software construct that has a defined use, has a run-time interface and can be autonomously deployed
- Appealing reasons behind OTS: economical, specialized, time to market….



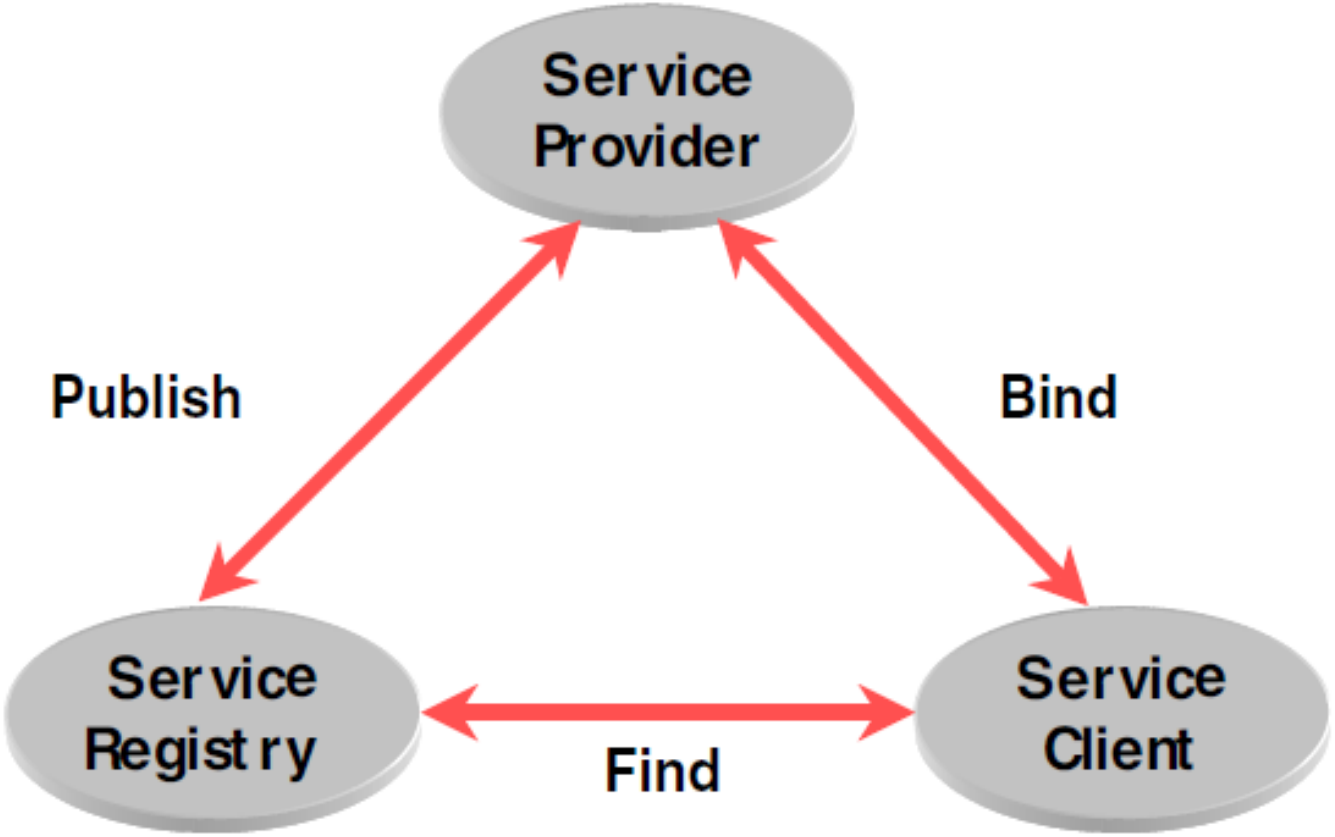
Service-Oriented Computing (SOC)

- SOC is the computing paradigm that utilizes services as fundamental elements for developing applications
  - Discrete pieces of software providing application functionality as services to other applications
- A service is a set of capabilities that is accessible via some kind of interface
  - Technology neutral
  - Loosely coupled
  - Support location transparency

Web Service

- Web service: a special case of SOC that use the Internet as the communication medium and open Internet-based standards
  - SOAP calls carries XML data content
  - WSDL expresses the service descriptions of the web-services using XML(ports, port types, bindings)
  - UDDI provides directory service containing service publications and enables web-service clients to locate candidate services and discover their details

The Basic Service Oriented Architecture (SOA)



System of Systems (SoS)

- Systems of systems are large-scale concurrent and distributed systems that are comprised of complex systems
  - A group of systems that are interoperating to achieve a joint purpose
  - Independent and useful systems are integrated into a larger system that delivers unique capabilities

Taxonomy of Systems of Systems

Type	Description
Directed	SoS objectives, centralized management, funding, and authority for the overall SoS are in place. Systems are subordinated to the SoS.
Acknowledged	SoS objectives, centralized management, funding, and authority in place. However, systems retain their own management, funding, and authority in parallel with the SoS
Collaborative	There are no overall objectives, centralized management, authority, responsibility, or funding at the SoS level. Systems voluntarily work together to address shared or common interests
Virtual	Like collaborative, but systems don't know about each other

Issues in Components/Services/Systems Interoperability

- Architectural issues
  - Enforce architectural assumptions
  - Compatibility issues
- Other issues
  - Vendor support issues
  - Configuration management
  - Documentation issues
  - Unknown technical features etc.

Architectural Mismatch

- Not all components/services/systems work together, problems might occur at different time
  - Compile, integration, run time
- Quality attributes usually not (fully) specified for components/services/systems
  - Then what are specified?
- Architectural mismatch
  - Reflects a mismatch between assumptions embodied in separately developed components

Interface Mismatch (1)

- A generalized definition of interface
  - The standard concept of interface in current practice: a component's API
  - The generalized concept concerns what we assume about interface in practical use: the assumptions that components can make about each other
    - side effects
    - consumption of global resources
    - coordination requirements...

Interface Mismatch (2)

- Architectural assumptions
  - Provides assumptions
    - the services a components/services/systems provides to its users or clients.
  - Requires assumptions
    - the services or resources that a components/services/ systems must have in order to correctly function.
- Architectural mismatch is treated as a special case of interface mismatch.

Techniques for Repairing Interface Mismatch

- Repairing the interface mismatch
  - Change the code of offending component
  - Use an alternative component
    - Sometimes not feasible options
  - Insert code for reconciliation
    - Wrappers
    - Bridges
    - Mediators

Wrappers

- Wrappers yield an alternative interface to the component, i.e. interface translation
  - Translating an element of an interface into an alternative element
  - Hiding an element of an interface
  - Preserving an element of a component's base interface without change

Bridges

- A bridge translates some requires assumptions of one arbitrary component to some provides assumptions of another.
- Difference between a bridge and a wrapper
  - the repair code constituting a bridge is independent of any particular component.
  - the bridge must be explicitly invoked by some external agent
  - bridges address specific assumptions, which means they focus on a narrower range of interface translations than wrappers

Mediators

- A mediator encapsulates how a set of objects interact
  - It incorporate a planning function that in effect results in runtime determination of the translation
    - Intelligence and dynamic behaviors can be implemented in mediators
  - e.g. high-fidelity data producer VS. low-fidelity data consumer VS. users with different throughput requirement

Interoperability (1)

- Interoperability is about the degree to which two or more systems can usefully exchange meaningful information via interfaces in a particular context
  - Syntactic interoperability: the ability to exchange data
  - Semantic interoperability: the ability to correctly interpret the data being exchanged

Interoperability (2)

- Two important aspects of interoperability
  - Discovery: the consumer of a service must discover the location, identity, and the interface of the service
  - Handling of the response:
    - The service reports back to the requester with the response
    - The service sends its response on to another system
    - The service broadcasts its response to any interested parties

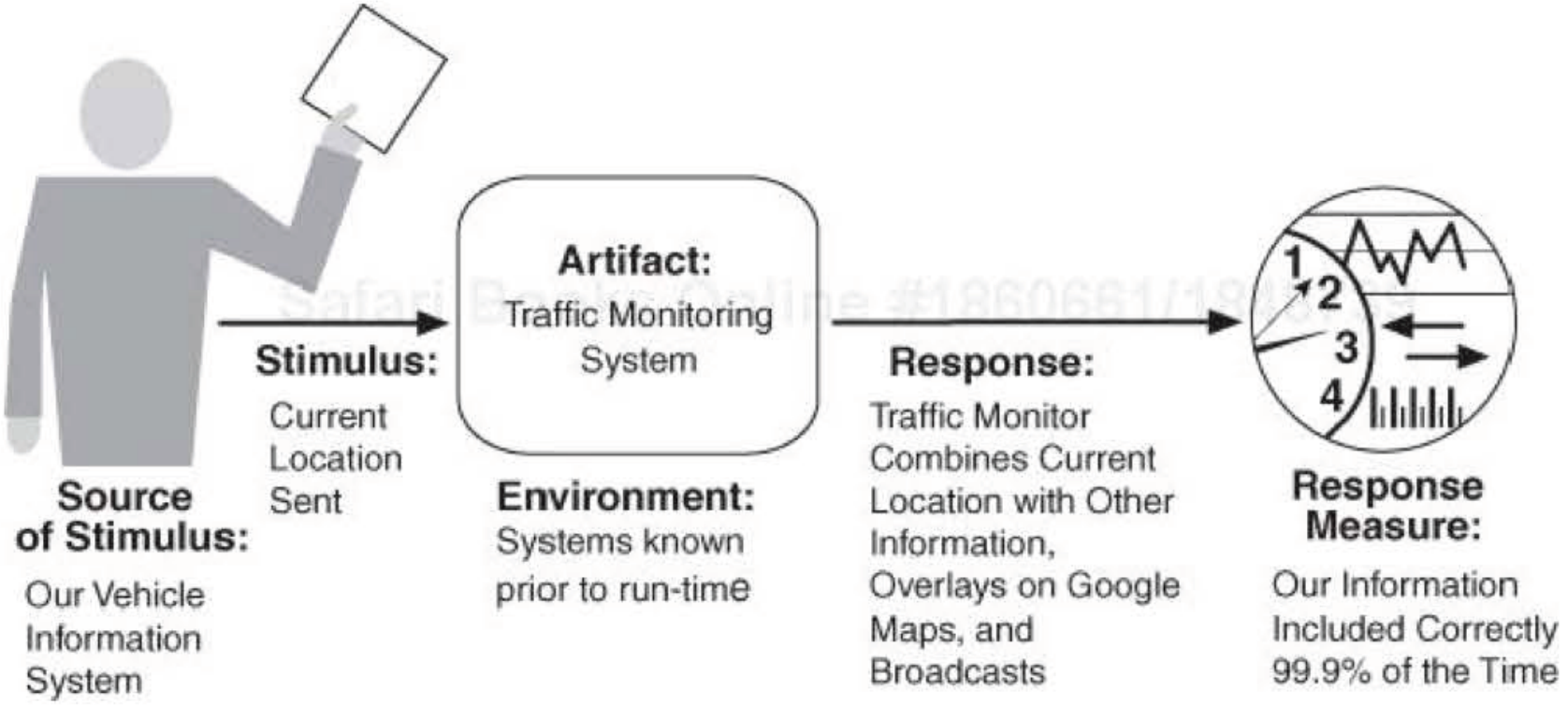
Interoperability General Scenario (1)

- Stimulus
  - A request to exchange information among system(s)
- Source of stimulus
  - A system initiates a request to interoperate with another system
- Response
  - The request is accepted and the information is understood by the receiving party both syntactically and semantically, or
  - the request is rejected and appropriate entities are notified
  - The request may be logged

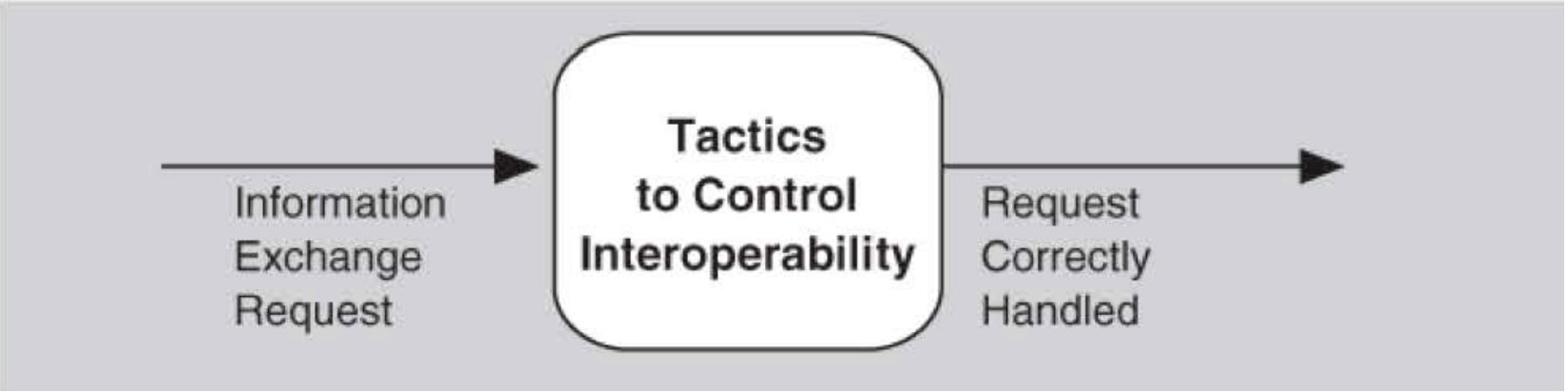
Interoperability General Scenario (2)

- Response measure
  - The percentage of information exchanges correctly processed or rejected
- Artifacts
  - The systems that wish to interoperate
- Environment
  - The systems that wish to interoperate are discovered at runtime or are known prior to runtime

A Sample Concrete Interoperability Scenario



Tactics for Interoperability



- Two categories of interoperability tactics
  - Locate
  - Manage interfaces

Locate: Discover Service

- Used when the systems that interoperate must be discovered at runtime
- Locate a service through searching a known directory service
  - May be multiple levels of indirection in this location process
  - The service can be located by type of service, by name, by location, or by some other attribute

Manage Interfaces: Orchestrate

- Coordinating, managing and sequencing the invocation of particular services
- Orchestration is used when the interoperating systems must interact in a complex fashion to accomplish a complex task
  - E.g. Workflow engines
  - The mediator design pattern can serve this function for simple orchestration
  - Complex orchestration can be specified in a language such as BPEL

Manage Interfaces: Tailor interface

- Adding or removing capabilities to an interface
  - Capabilities such as translation, adding buffering, or smoothing data can be added
  - Capabilities may be removed as well
    - E.g. removing capabilities is to hide particular functions from untrusted users

Architectural Design Support for Interoperability

- We check the architectural design and analysis process for interoperability from the following aspects:
  1. Allocation of responsibilities
  2. Coordination model
  3. Data model
  4. Management of resources
  5. Mapping among architectural elements
  6. Binding time decisions
  7. Choice of technology

Allocation of Responsibilities

- The following system responsibilities are to be allocated to ensure system interoperability
  - Detect a request to interoperate with known or unknown external systems
  - Accept the request
  - Exchange information
  - Reject the request
  - Notify appropriate entities (people or systems)
  - Log the request (support for nonrepudiation)

Coordination Model

- Ensure that the coordination mechanisms can meet the critical quality attribute requirements
- Special considerations for performance:
  - Network traffic
  - Jitter in network transmission
  - ...

Data Model

- Determine the syntax and semantics of the major data abstractions that may be exchanged among interoperating systems
- Ensure that these major data abstractions are consistent with data from the interoperating systems

Mapping among Architectural Elements

- Focus on components → processors mapping
  - Ensure communication among interoperating components
  - Ensure the communication meet other quality requirements

Resources Management

- Ensure the interoperation with other systems will not exhaust critical system resource
- The resource load imposed by the communication requirements of interoperation shall be acceptable
- Arbitration policy for resource contention among participating systems

Choice of Technology

- For any of your chosen technologies, are they "visible" at the interface boundary of a system?
- Consider the quality attributes of technologies that are used in your design, such as web services.

Reading Assignment

- Read Chapter 7 of the textbook.