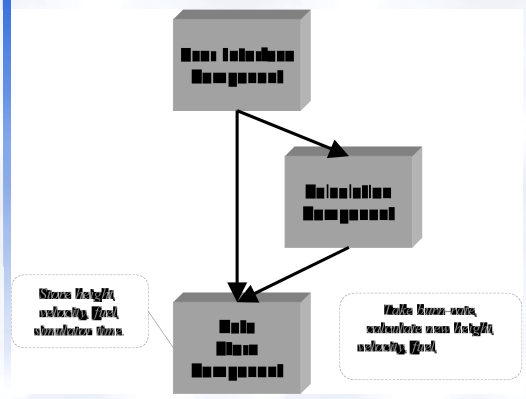
Analysis of Software Architectures

Software Architecture
Lecture 13

What Is Architectural Analysis?

- Architectural analysis is the activity of discovering important system properties using the system's architectural models.
  - Early, useful answers about relevant architectural aspects
  - Available prior to system's construction
- Important to know
  1. which questions to ask
  2. why to ask them
  3. how to ask them
  4. how to ensure that they can be answered

Informal Architectural Models and Analysis



- Helps architects get clarification from system customers
- Helps managers ensure project scope
- Not as useful to developers

Formal Architectural Models and Analysis

```
Component UserInterface
  Port getValues
  Port calculate
  Computation
Connector Call
  Role Caller =
  Role Callee =
  Glue =
Configuration LunarLander
  Instances
    DS : DataStore
    C : Calculation
    UI : UserInterface
    CtoUIgetValues, CtoUIstoreValues, UItoC, UItoDS : Call
  Attachments
    C.getValues as CtoUIgetValues.Caller
    DS.getValues as CtoUIgetValues.Callee
    C.storeValues as CtoUIstoreValues.Caller
    DS.storeValues as CtoUIstoreValues.Callee
    UI.calculate as UItoC.Caller
    C.calulate as UItoC.Callee
    UI.getValues as UItoDS.Caller
    DS.getValues as UItoDS.Callee
End LunarLander.
```

- Helps architects determine component composability
- Helps developers with implementation-level decisions
- Helps with locating and selecting appropriate OTS components
- Helps with automated code generation
- Not as useful for discussions with non-technical stakeholders

Concerns Relevant to Architectural Analysis

- Goals of analysis
- Scope of analysis
- Primary architectural concern being analyzed
- Level of formality of architectural models
- Type of analysis
- Level of automation
- System stakeholders interested in analysis

Architectural Analysis Goals

- The four "C"s
  - Completeness
  - Consistency
  - Compatibility
  - Correctness

Architectural Analysis Goals – Completeness

- Completeness is both an external and an internal goal
- It is *external* with respect to system requirements
  - Challenged by the complexity of large systems' requirements and architectures
  - Challenged by the many notations used to capture complex requirements as well as architectures
- It is *internal* with respect to the architectural intent and modeling notation
  - Have all elements been fully modeled in the notation?
  - Have all design decisions been properly captured?

Architectural Analysis Goals – Consistency

- Consistency is an internal property of an architectural model
- Ensures that different model elements do not contradict one another
- Dimensions of architectural consistency
  - Name
  - Interface
  - Behavior
  - Interaction
  - Refinement

Name Consistency

- Component and connector names
- Component service names
- May be non-trivial to establish at the architectural level
  - Multiple system elements/services with identical names
  - Loose coupling via publish-subscribe or asynchronous event broadcast
  - Dynamically adaptable architectures

Interface Consistency

- Encompasses name consistency
- Also involves parameter lists in component services
- A rich spectrum of choices at the architectural level
- Example: matching provided and required interfaces

```
ReqInt:   getSubQ(Natural first, Natural last, Boolean remove)
          returns FIFOQueue;

ProvInt1: getSubQ(Index first, Index last)
          returns FIFOQueue;

ProvInt2: getSubQ(Natural first, Natural last, Boolean remove)
          returns Queue;
```

Behavioral Consistency

- Names and interfaces of interacting components may match, but behaviors need not
- Example: subtraction

```
subtract(Integer x, Integer y) returns Integer;
```

  - Can we be sure what the *subtract* operation does?
- Example: QueueClient and QueueServer components

QueueClient

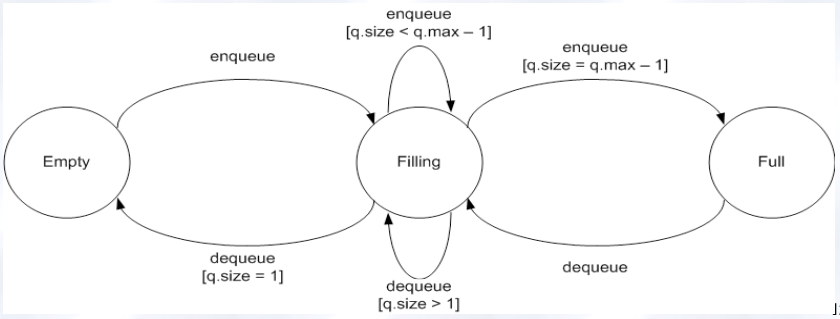
```
precondition  q.size > 0;
postcondition ~q.size = q.size;
```

QueueServer

```
precondition  q.size > 1;
postcondition ~q.size = q.size - 1;
```


Interaction Consistency

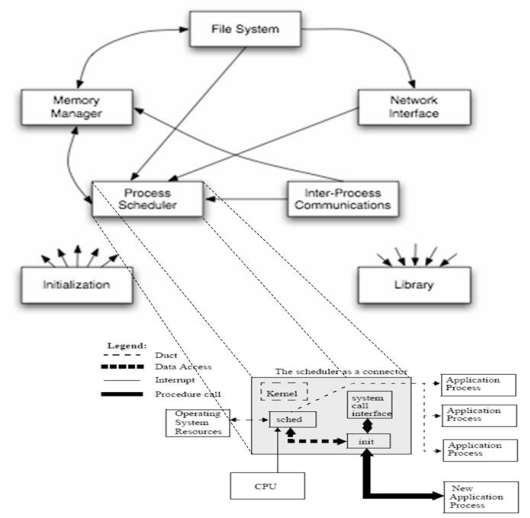
- Names, interfaces, and behaviors of interacting components may match, yet they may still be unable to interact properly
- Example: QueueClient and QueueServer components



Refinement Consistency

- Architectural models are refined during the design process
- A relationship must be maintained between higher and lower level models
  - All elements are preserved in the lower level model
  - All design decisions are preserved in the lower-level model
  - No new design decisions violate existing design decisions

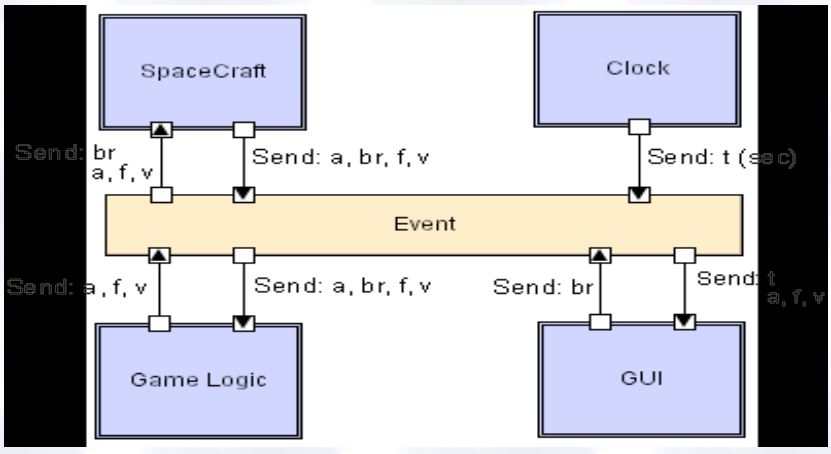
Refinement Consistency Example



Architectural Analysis Goals – Compatibility

- Compatibility is an external property of an architectural model
- Ensures that the architectural model adheres to guidelines and constraints of
  - a style
  - a reference architecture
  - an architectural standard

Architectural Compatibility Example – Lunar Lander



Architectural Analysis Goals – Correctness

- Correctness is an external property of an architectural model
- Ensures that
  1. the architectural model fully realizes a system specification
  2. the system's implementation fully realizes the architecture
- Inclusion of OTS elements impacts correctness
  - System may include structural elements, functionality, and non-functional properties that are not part of the architecture
  - The notion of *fulfillment* is key to ensuring architectural correctness

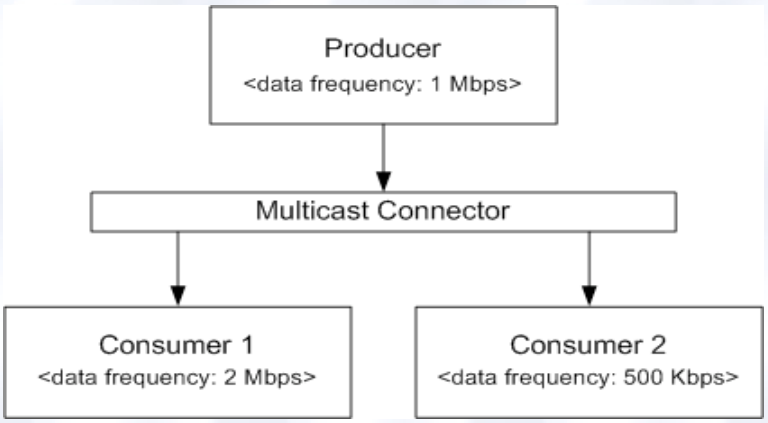
17

Scope of Architectural Analysis

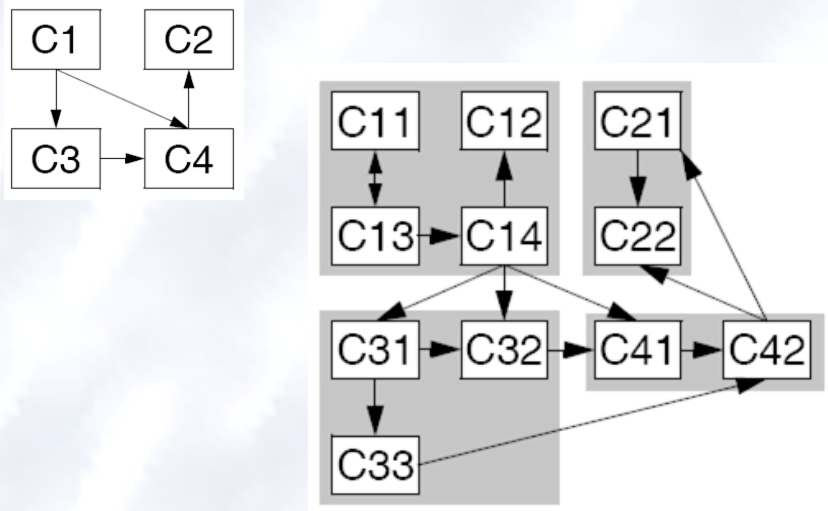
- Component- and connector-level
- Subsystem- and system-level
  - Beware of the "honey-baked ham" syndrome
- Data exchanged in a system or subsystem
  - Data structure
  - Data flow
  - Properties of data exchange
- Architectures at different abstraction levels
- Comparison of two or more architectures
  - Processing
  - Data
  - Interaction
  - Configuration
  - Non-functional properties

18

Data Exchange Example



Architectures at Different Abstraction Levels



Architectural Concern Being Analyzed

- Structural characteristics
- Behavioral characteristics
- Interaction characteristics
- Non-functional characteristics

Level of Formality

- Informal models
- Semi-formal models
- Formal models

Type of Analysis

- Static analysis
- Dynamic analysis
- Scenario-driven analysis
  - Can be both static and dynamic

Level of Automation

- Manual
- Partially automated
- Fully automated

Analysis Stakeholders

- Architects
- Developers
- Managers
- Customers
- Vendors

Architectural Analysis in a Nutshell