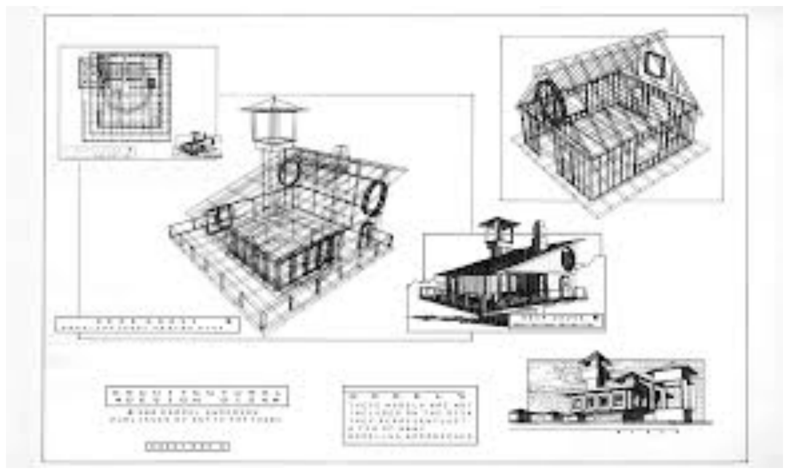


COMP 3004

Introduction to Software Architecture

Winter 2020

Instructor: Dr. Olga Baysal

Carleton
UNIVERSITY

Material and slides from:
Mike Godfrey

Topics

- **Software Architecture**
- **Software Architecture’s Elements**

Architecture

- Architecture is:
  - All about communication
  - What “parts” are there?
  - How do the “parts” fit together?

- Architecture is not:
  - About development
  - About algorithms
  - About data structures

AA 550DT

Software Architecture

- Definition:
- "Software architecture is **the set of principal design decisions governing a system**"
- Blueprint for construction and evolution.
- Design decisions encompass every facet of the system under development:
  - Structure
  - Behaviour
  - Interaction
  - Non-functional properties

Other Definitions

- **SEI [Garlan and Shaw]:**

  “The software architecture of a program or computing system is the **structure** or structures of the system, which comprise software **elements**, the externally visible **properties** of those elements, and the **relationships** among them”

Other Definitions

- **ANSI/IEEE 1472-200:**

  "Architecture is the **fundamental organization** of a system, embodied in its **components**, their **relationships** to each other and the environment, and the principles governing its design and evolution"

Other Definitions

- **Eoin Woods:**

  "Software architecture is the set of design decisions which, if made incorrectly, may cause your project to be cancelled"

So What?

- What makes building systems so hard?
  - Young field with high user expectations
  - Building of complex but intangible systems
  - Software cannot execute independently
- **Accidental difficulties** [Brooks]
  - Problems that can be overcome (e.g., ...)
- **Essential difficulties** [Brooks]
  - Those problems that cannot be easily overcome
  - Complexity, conformity, intangibility, changeability

Analogy: Architecture of Buildings

- We live in them
- We know (approximately) how they are built
  - Requirements
  - Design (blueprints)
  - Construction
  - Use in practice
- This is similar (though not identical) to how we build software

Some Parallels

- Satisfaction of customers' needs
- Specialization of labor
- Intermediate points where plans and progress are reviewed
- Architecture is different from, but linked with the product/structure
- Properties of structures are induced by the design of the architecture
- The architect has a distinctive role and character

The Architect

- A distinctive role and character in a project
- Very broad training
- A keen sense of aesthetics
- Deep understanding of the domain
  - Properties of structures, materials, and environments
  - Needs of customers

Exercises

- How is building architecture **different** from software architecture?
- What **common benefits** can software gain from an architect that a building gets from its architect?

Limitations of Analogy

- We know a lot about buildings, much less about software
- The nature of software is different from that of building architecture
- Software is much more malleable than physical materials
- The two “construction industries” are very different
- Software deployment has no counterpart in building architecture
- Software is a machine; a building is not

… Yet the Power of Architecture

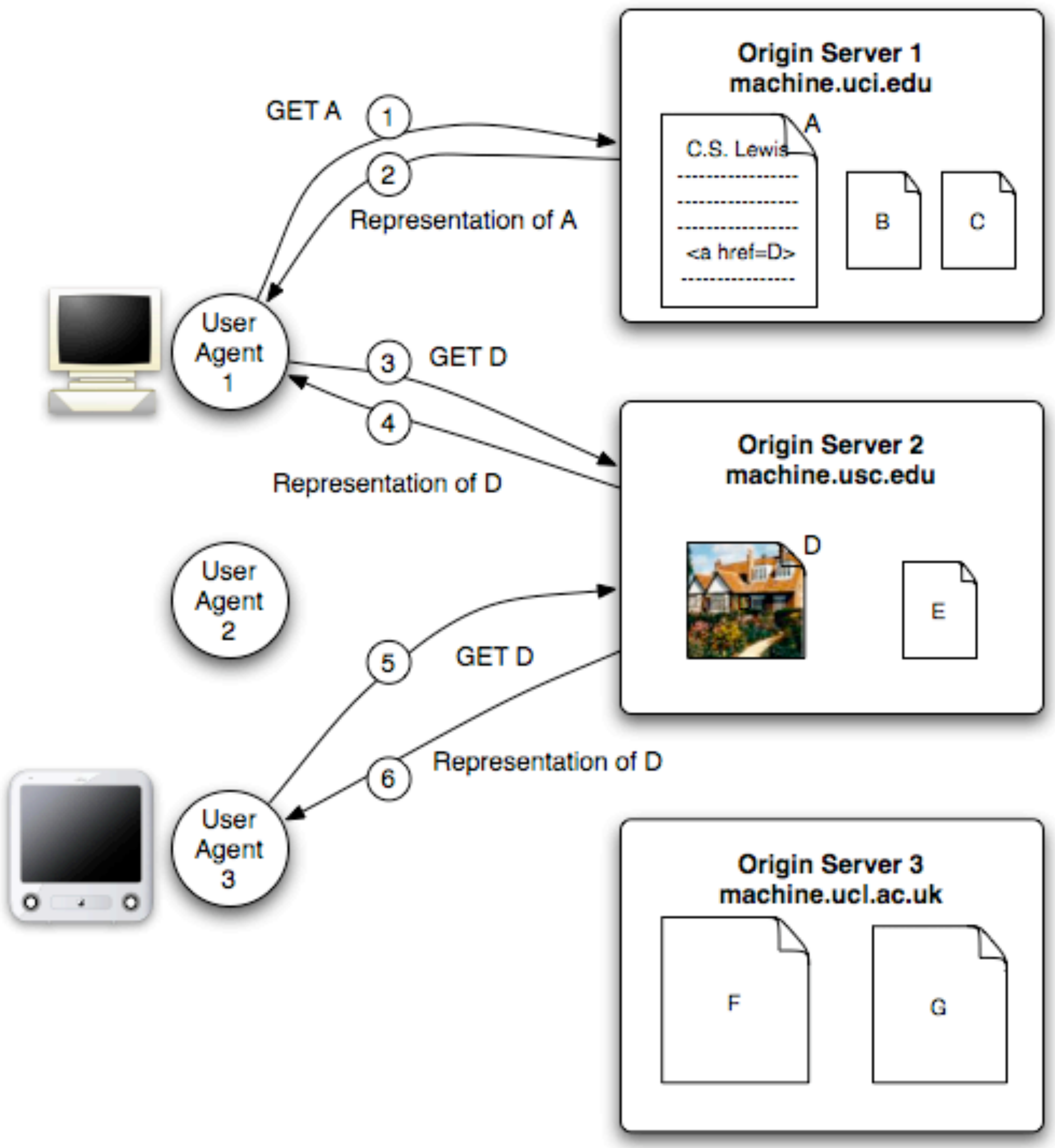
- Intellectual control
- Conceptual integrity
- Effective basis for knowledge reuse
- Realizing experience, designs, and code
- Effective project communication
- Management of a set of variant systems

WWW Example

WWW Example

WWW Example

- And this



WWW in a Nutshell

- The Web is a collection of resources, each of which has a unique name known as a “URL”
- Each resource denotes, informally, some information
- URL’s can be used to determine the identity of a machine on the Internet, known as an origin server, where the value of the resource may be ascertained
- Communication is initiated by clients, known as user agents, who make requests to servers
  - Web browsers are common instances of user agents

WWW in a Nutshell

- Resources can be manipulated through their representations
  - HTML is a very common representation language used on the Web
- All communication between user agents and origin servers must be performed by a simple, generic protocol (HTTP), which offers the command methods GET, POST, etc.
- All communication between user agents and origin servers must be fully self-contained (so-called “stateless interactions”)

WWW Architecture

- Architecture of the Web is wholly separate from the code
- There is no single piece of code that implements the architecture
- There are multiple pieces of code that implement the various components of the architecture
    - E.g., different web browsers

Architecture Views

- A software architecture is a complex design artifact

- Many possible “views” of the architecture
  - Cf. with buildings – floor plan, external, electrical, plumbing, air-conditioning

Temporal Aspects

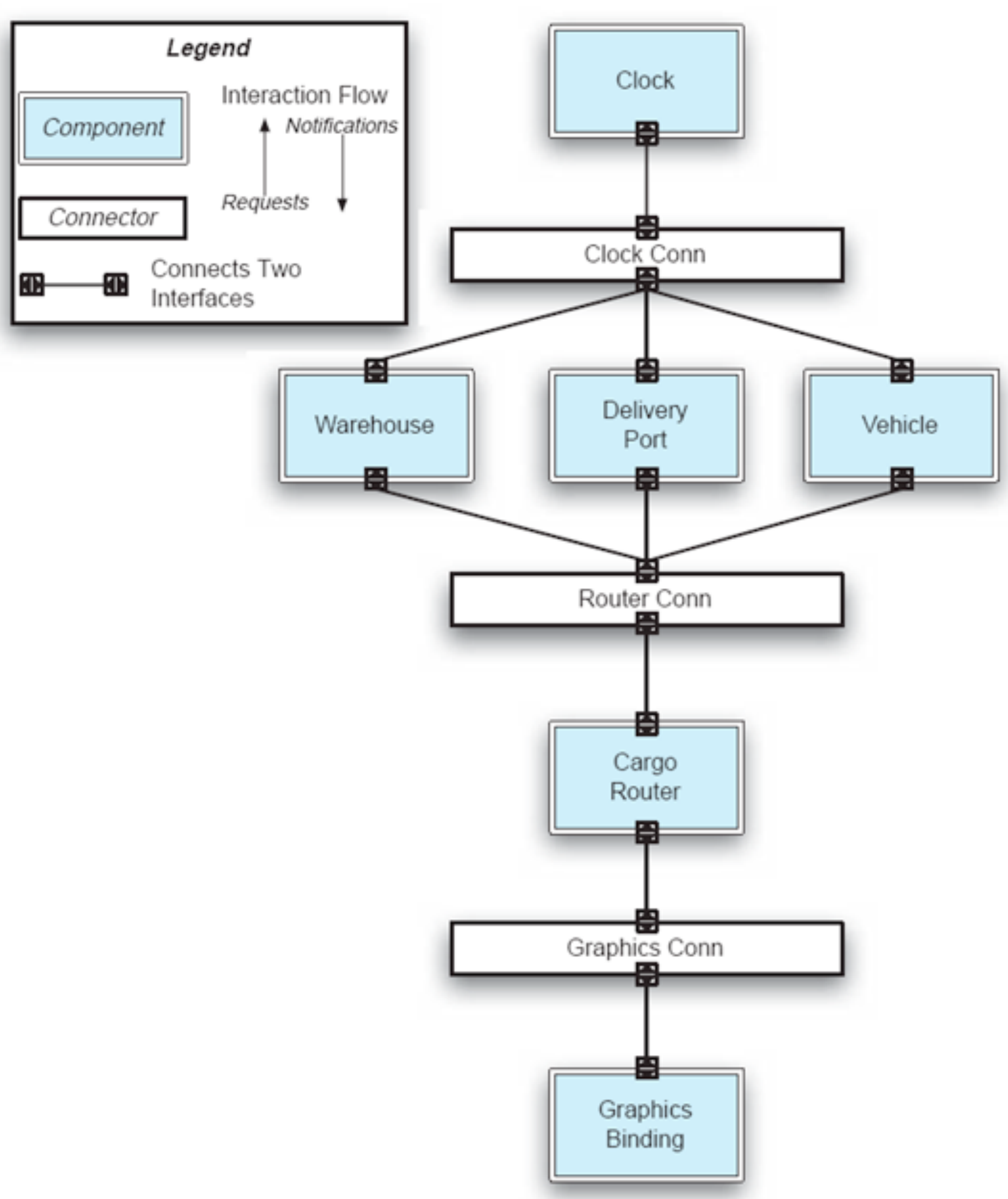
- A software architecture is a complex design artifact
- Observation: Design decisions are made and unmade over a system's lifetime
    - Consequence: architecture has a **temporal** aspect
- At any given point in time a system has only one architecture
    - Architecture will change over time

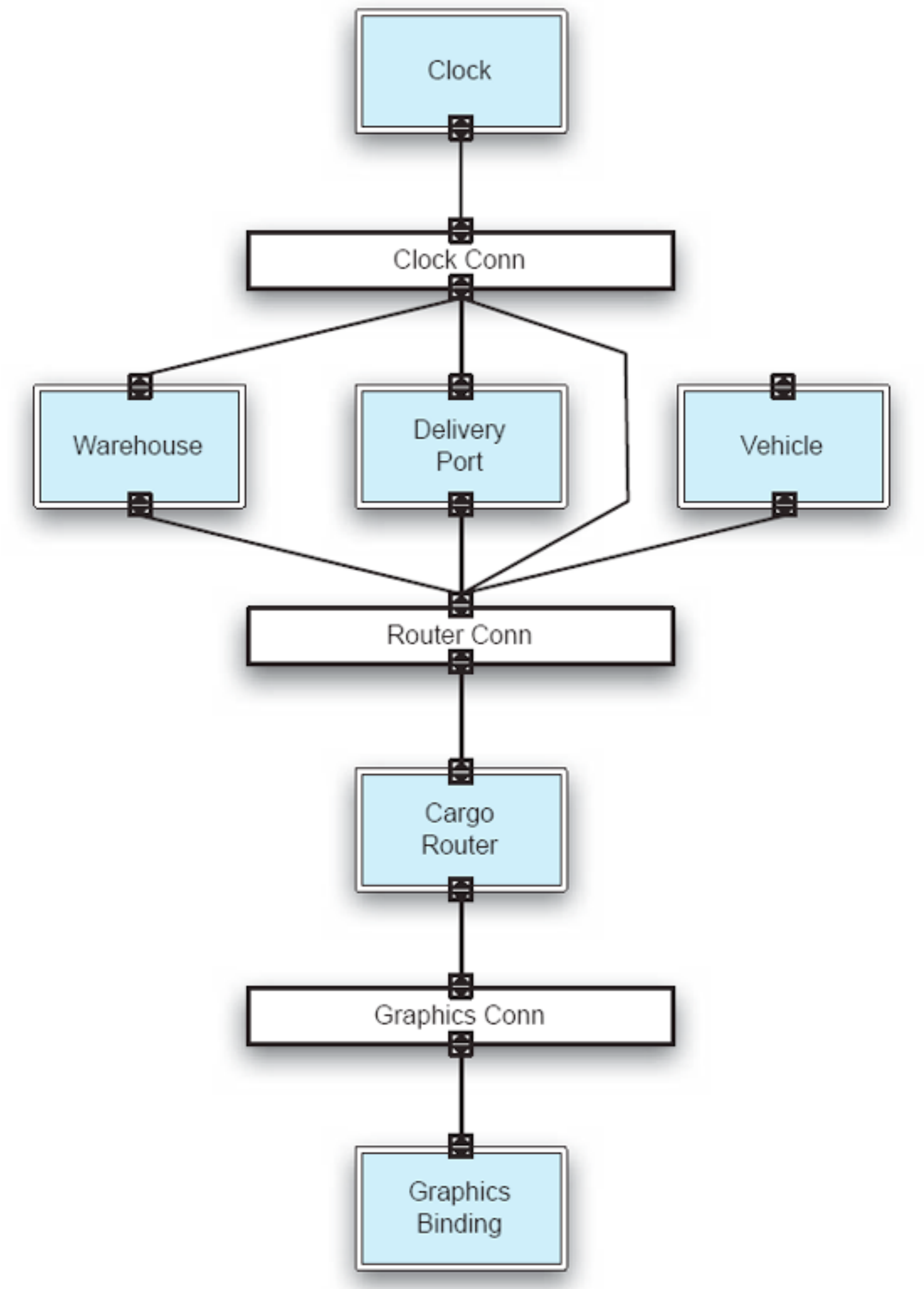
Prescriptive vs. Descriptive

- **Prescriptive architecture**: dictates how the system will be built a priori
  - *as-conceived* or *as-intended* architecture

- **Descriptive architecture** describes how the system has been built
  - *as-implemented* or *as-realized* architecture

As-Designed vs. As-Implemented Architecture





[Taylor et al. Software Architecture: Foundations, Theory and Practice.]

Architectural Evolution

- **Ideally,** its prescriptive architecture is modified first, when a system evolves
- **In practice,** the system – and thus its descriptive architecture – is often directly modified
- Reasons:
  - Developer sloppiness
  - Perception of short deadlines which prevent thinking through and documenting
  - Lack of documented prescriptive architecture
  - Need or desire for code optimizations
  - Inadequate techniques or tool support

Architectural Degradation

- **Architectural drift** : introduction of principal design decisions into the descriptive architecture that
  - are not included in, encompassed by, or implied by the prescriptive architecture
  - but which do not violate any of the prescriptive architecture's design decisions
  - (new constraints are introduced, without violating the descriptive architecture)

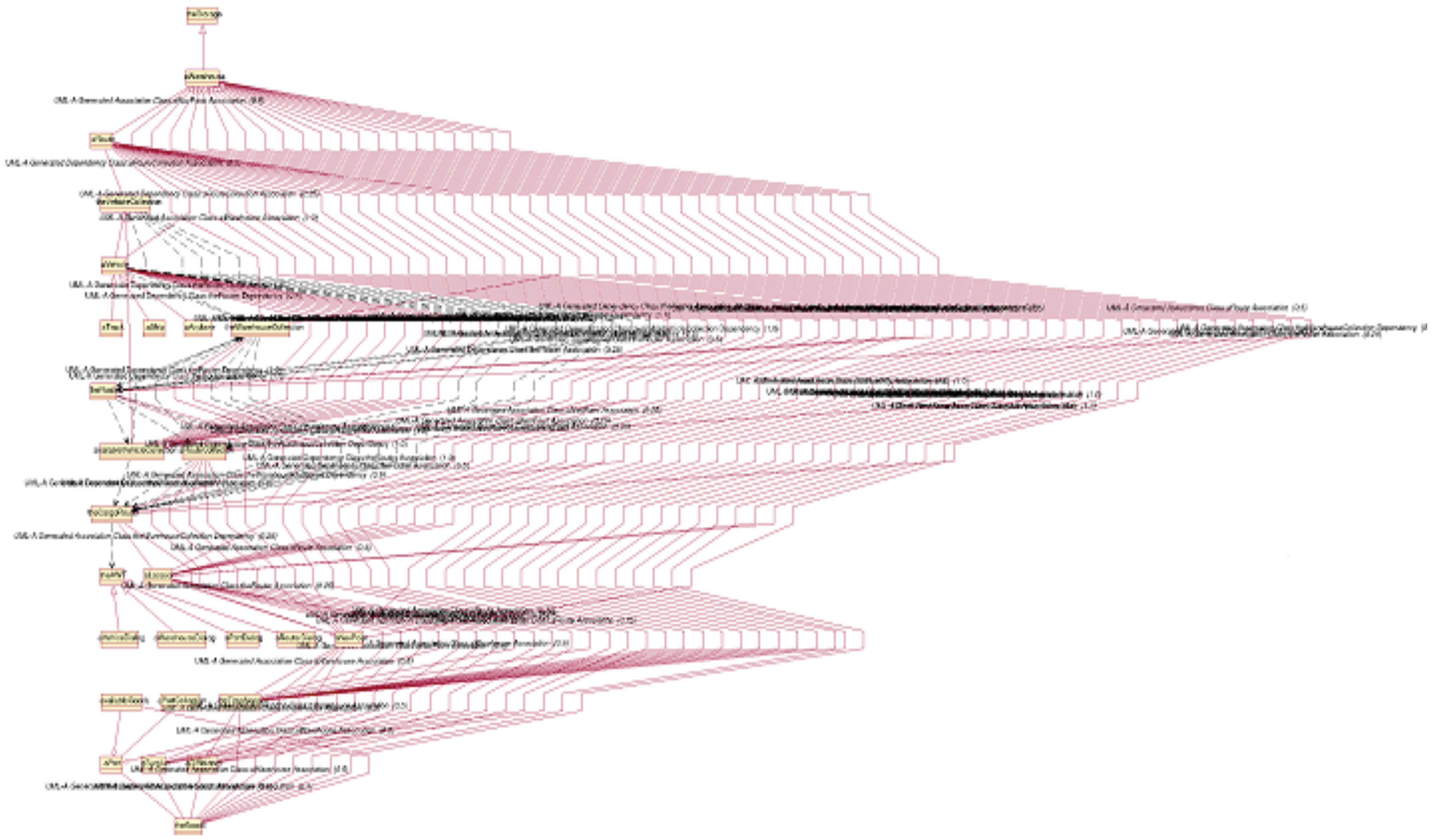
- **Architectural erosion** is the introduction of architectural design decisions into a system's descriptive architecture that violate its prescriptive architecture

Architectural Recovery

- If architectural degradation is allowed to occur, one will be forced to recover the system's architecture sooner or later

- **Architectural recovery**: determining a software system's architecture from its implementation-level artifacts
    - Source code, executables, deployment

Implementation-Level View



[Taylor et al. Software Architecture: Foundations, Theory and Practice.]

Software Architecture’s Elements

- A software system’s architecture generally involves composition and interplay of different elements
  - **Processing** (may be referred as functionality or behaviour)
  - **Data** (also referred as information or state)
  - **Interaction**

Components

- Elements that encapsulate processing and data in a system's architecture are referred to as **software components**
- **Definition:**

  A software component is an architectural entity that

  - encapsulates a subset of the system's functionality
  - restricts access via explicit interface
  - has explicit environmental dependencies
- Components typically provide application-specific services

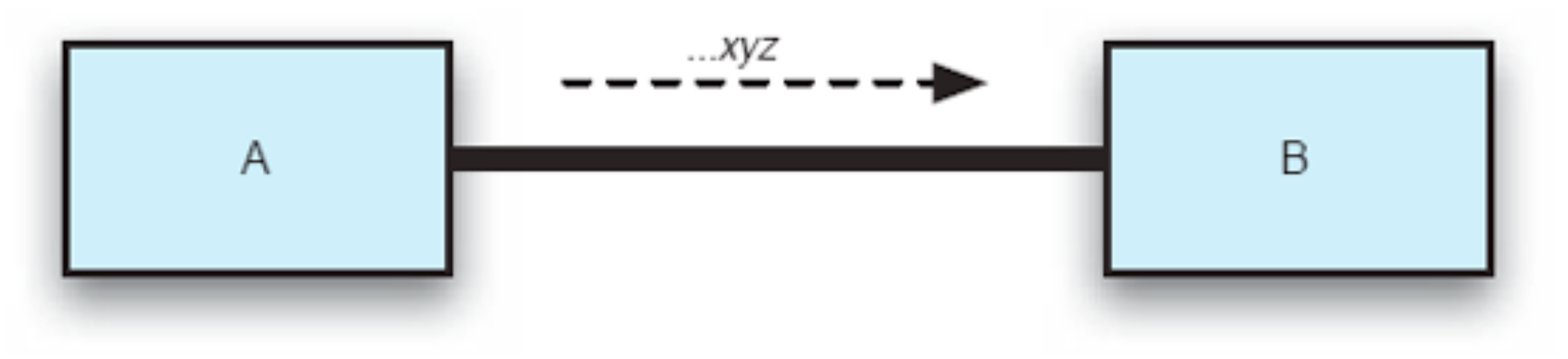
Connectors

- **Definition:**

  An architectural entity tasked with effecting and regulating interactions among components

- In many systems connectors are usually simple procedure calls or shared data accesses
  - Examples: …
- Often provide application-independent interaction mechanisms

Connectors in Action: Example



[Taylor et al. Software Architecture: Foundations, Theory and Practice.]

Configuration

- Bind components and connectors together in a specific way
- Definition:

An **architectural configuration**, or topology, is a set of specific associations between the components and the connectors of the system's architecture

- Differentiates a bag of components and connectors from an implementable system

Summary

- Software is complex
- So are buildings
  - And other engineering artifacts
  - Building architectures are an attractive source of analogy
- Software engineers can learn from other domains
- They also need to develop — and have developed — a rich body of their own architectural knowledge and experience

Next Class

- **Architectural Styles**