

## Building Architecture

- Architecture descriptions
  - Vocabulary
    - 2-bedroom, 1 bathroom, 1 living & dinner & kitchen, 1 garage, etc.
    - Cost, space, location, community, school, shopping, etc.
  - Languages
    - Informal, e.g., text, diagrams
    - Formal
  - Architecture styles (patterns or idioms)
    - Apartment, house, townhouse, condominium
    - Victorian, cathedral, pyramidal
    - Modern, ancient

108

## Software Architecture

- Architecture descriptions
  - Vocabulary
    - Components, connectors, properties
  - Languages
    - Informal, e.g. diagrams
    - Formal, e.g. architecture description languages (ADLs)
  - Styles, patterns or idioms
    - Client-server, RPC, Object-oriented, etc.

109

## ADLs

- Aesop
- Adage
- Meta-H
- C2
- Rapide
- SADL
- UniCon
- Wright

## Common Architecture Styles

- Data flow systems
  - Batch sequential
  - Pipe & Filter
- Call-and-return systems
  - Main program & subroutine
  - OO systems
  - Hierarchical layers
- Independent components
  - Communicating processes
  - Event systems
- Virtual machines
  - Interpreters
  - Rule-based systems
- Data-centered systems
  - Databases
  - Hypertext systems
  - Blackboards
- Process-control
- Client-server systems

## Architecture Descriptions

- “Camelot is based on the **client-server model** and uses **remote procedure calls** both locally and remotely to provide communication among applications and servers”
  - What is client? What’s a client like?
    - Applications, e.g., a terminal emulator, web browser, a bank teller
  - What is server? What’s a server like?
    - Centralized processes, e.g. database, file server
  - Why client-server model?
    - Distributed data, processing,
  - What is the communication mechanism?
    - Client stub, server stub
  - What is communicated? (data? control? process?)
  - Why RPC? Why not socket?
  - Any constraint?
    - Number of client? Authentication & authorization? Reliability? Availability?

112

## Software Architecture in Practice

- Software architectures are indeed used very often without being noticed
  - Good software developers have often adopted one or several architectural patterns informally and often implicitly
  - It is natural to use it
- Carries some information, but not a lot
  - Care must be taken
- No explicit description of the structure
  - No clear basis for communication and reasoning

113

## A Model of Software Architecture

- **Elements** (components/parts) from which systems are built, e.g., process, data, object, agent
- **Interactions** (connections/connectors/glues/relationships) between the elements, e.g. RPC, client-server
- **Patterns** describing layout of elements and interactions, e.g., # of element/connectors, order, topology, directionality
- **Constraints** on the patterns, e.g., temporal, cardinality, concurrency, (a)synchronous
- **Styles**: abstraction of architectural components recurring in various specific architectures
- **Rationale**: describe why the particular architecture is chosen