Lecture 27:
Introduction to Software Architecture

Kenneth M. Anderson

Foundations of Software Engineering

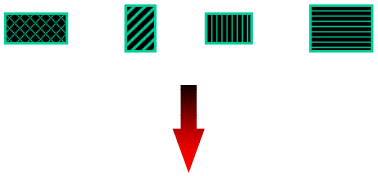
CSCI 5828 - Spring Semester, 1999

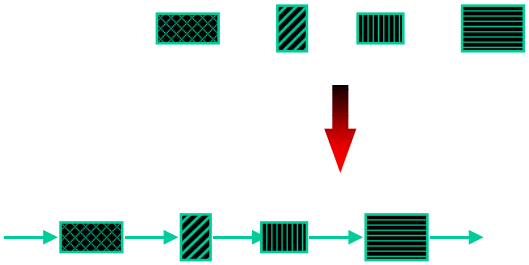
Today's Lecture

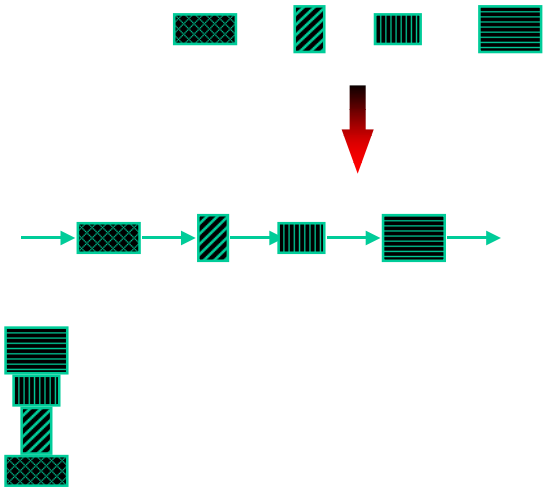
- Introduction and Background of
  - Software Architecture
    - concepts
    - styles
    - domains

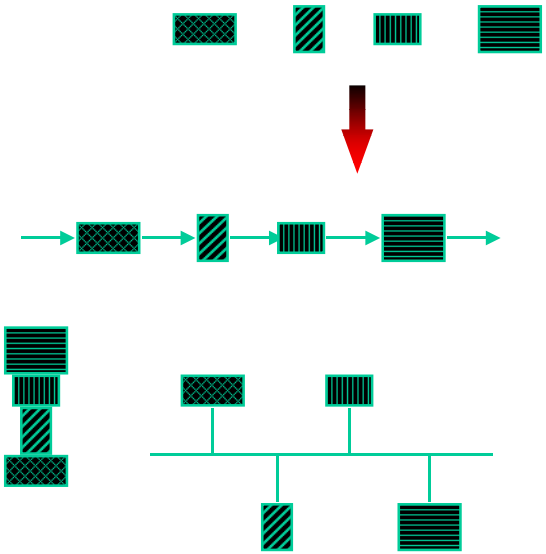
Software Architecture

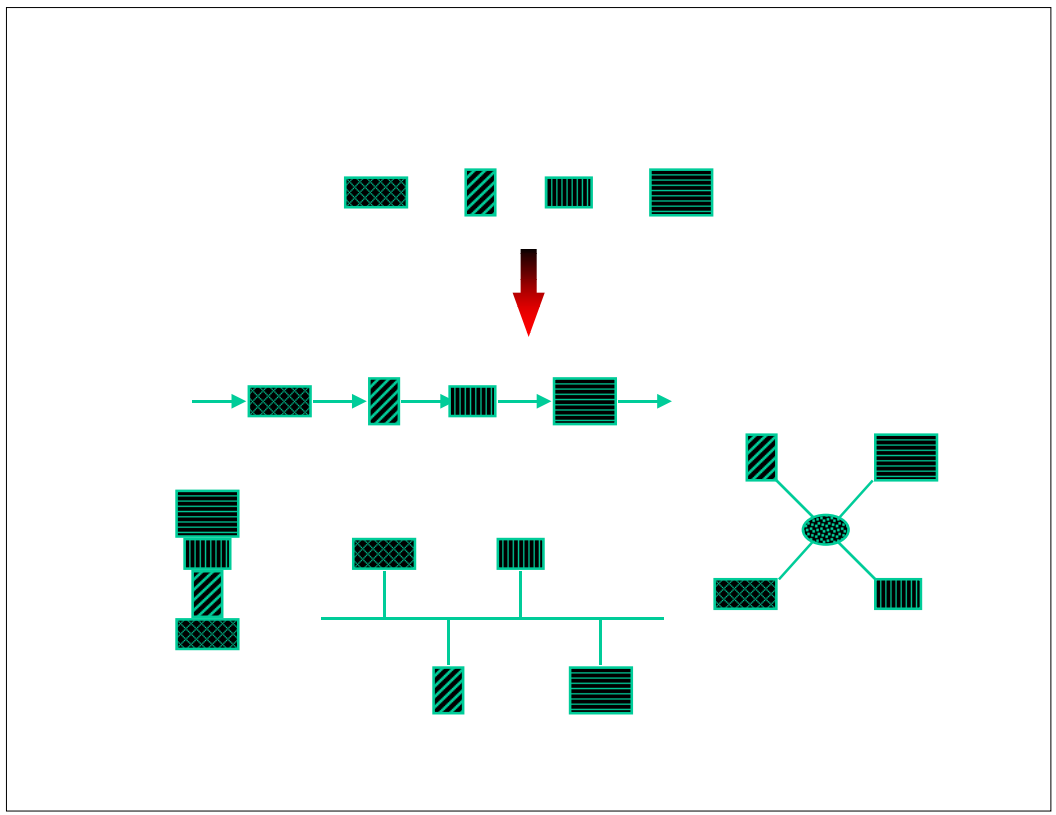
- The principled study of software components, including their properties, relationships, and patterns of combination
- Also, a particular set of software components as combined in a particular software system

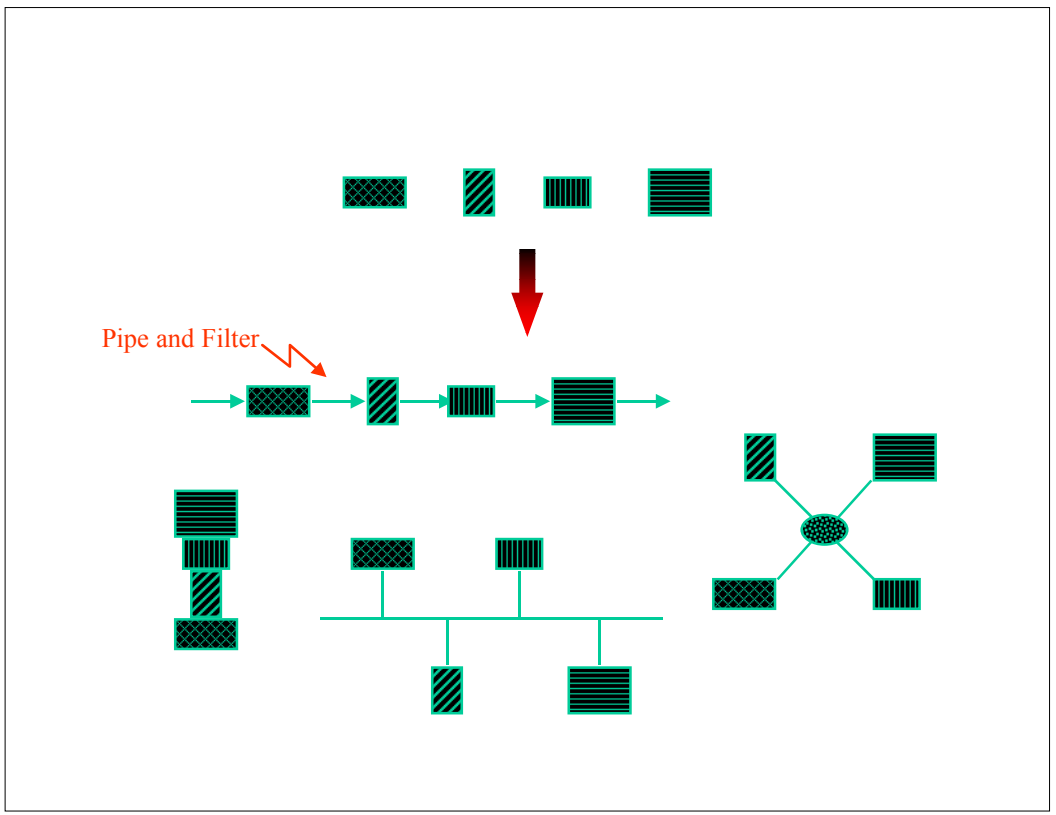


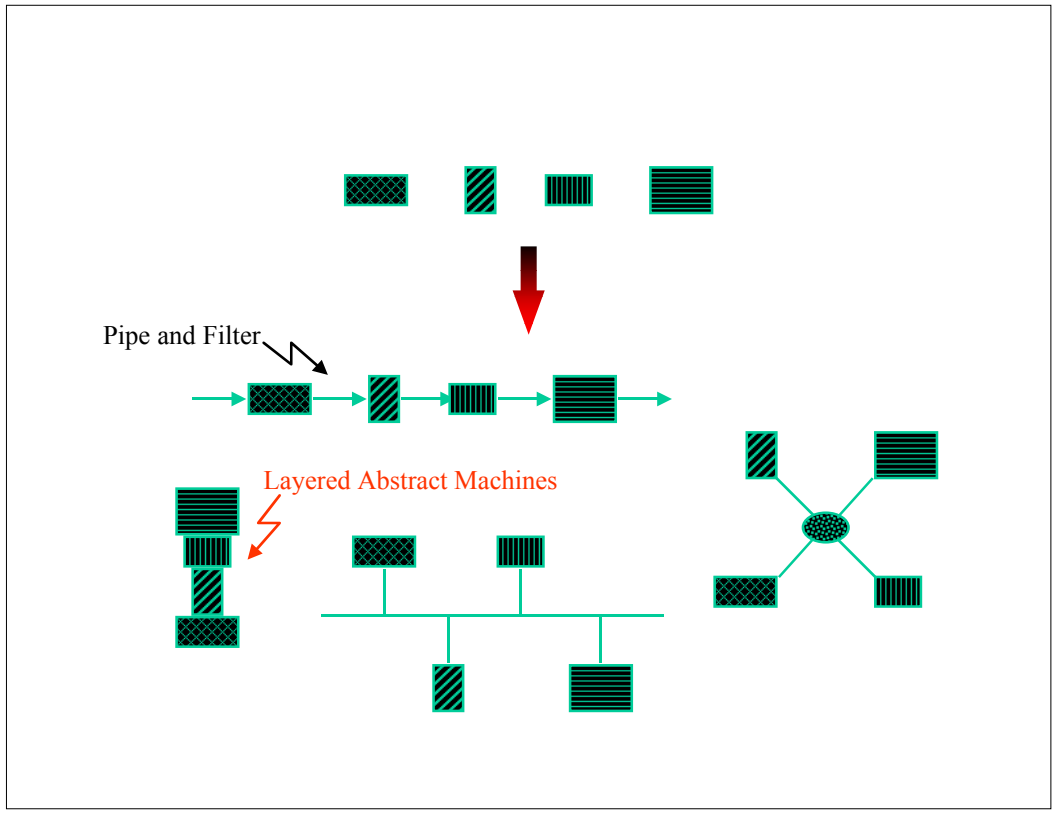


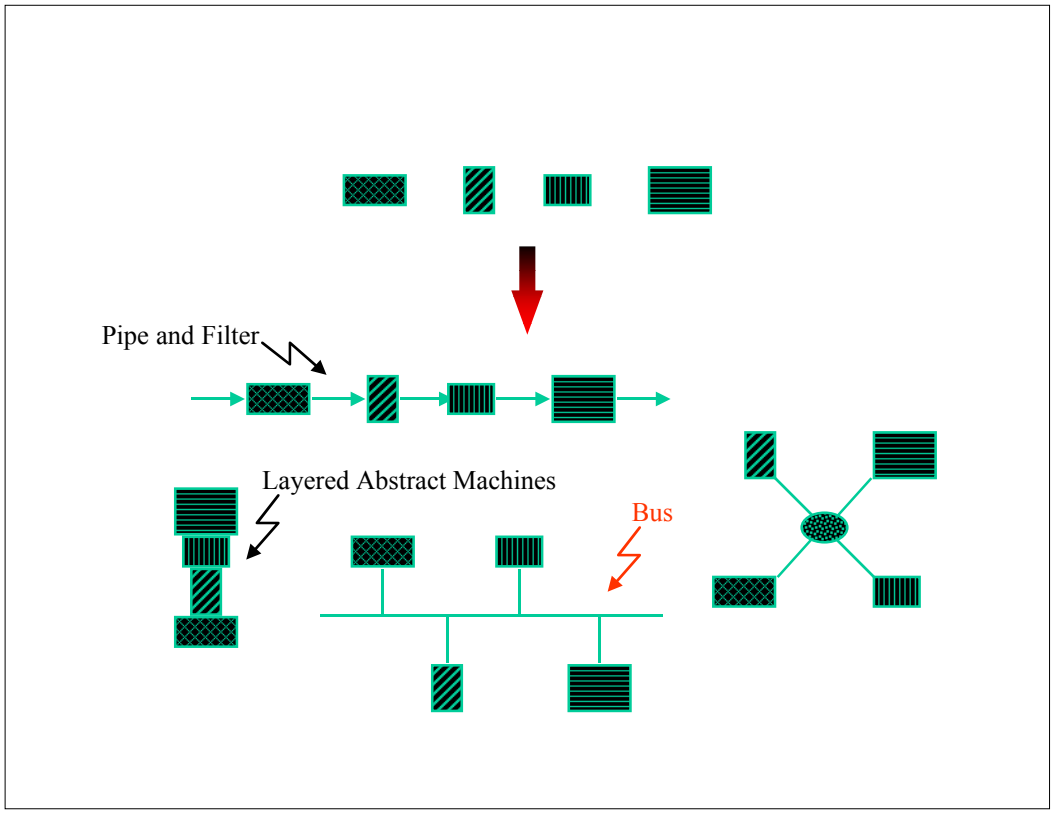


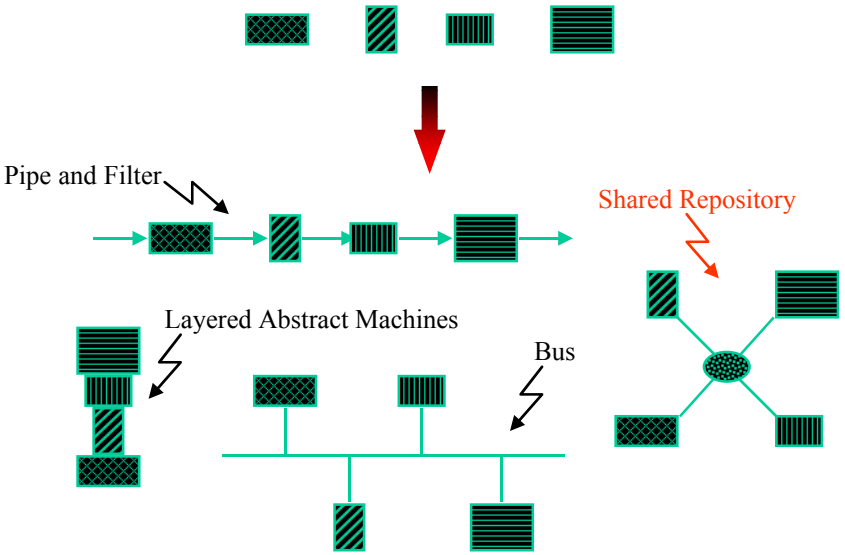




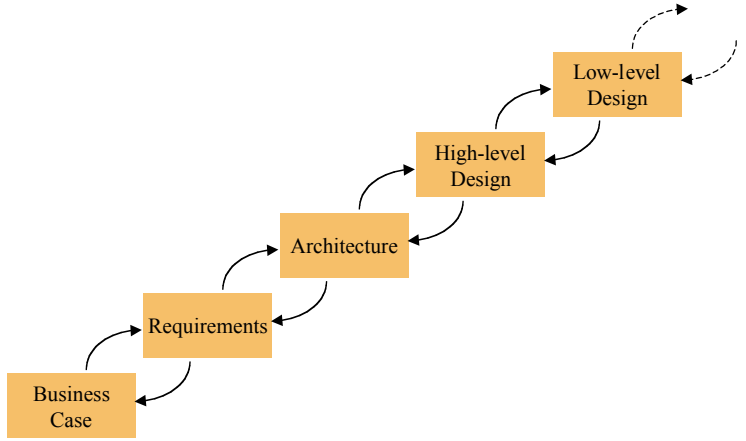

Pipe and Filter


Pipe and Filter
Layered Abstract Machines

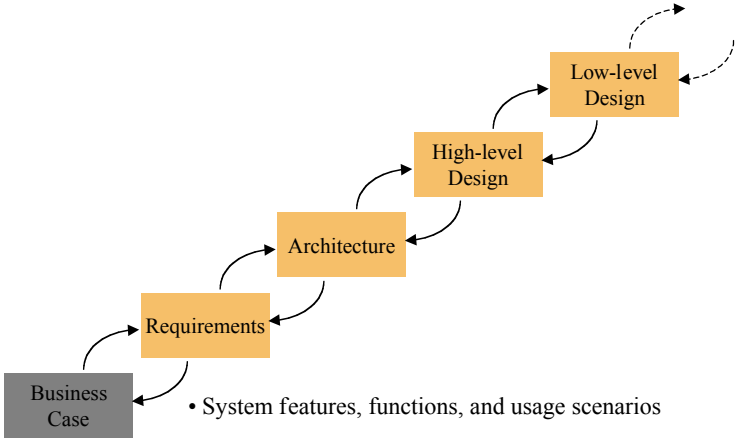

Pipe and Filter
Layered Abstract Machines
Bus



The Role of Architecture

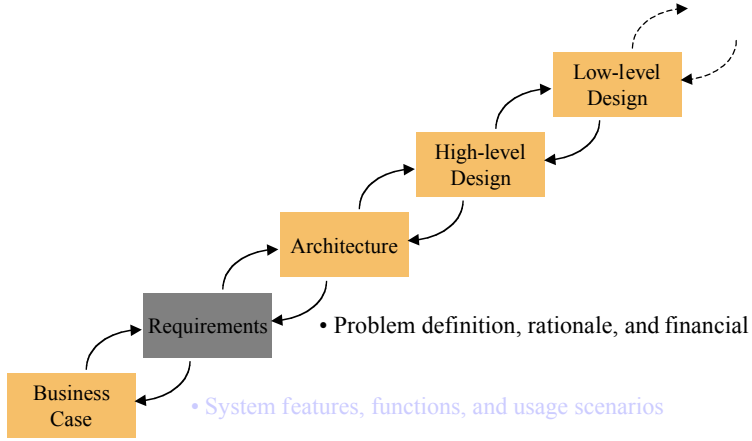


The Role of Architecture



• System features, functions, and usage scenarios

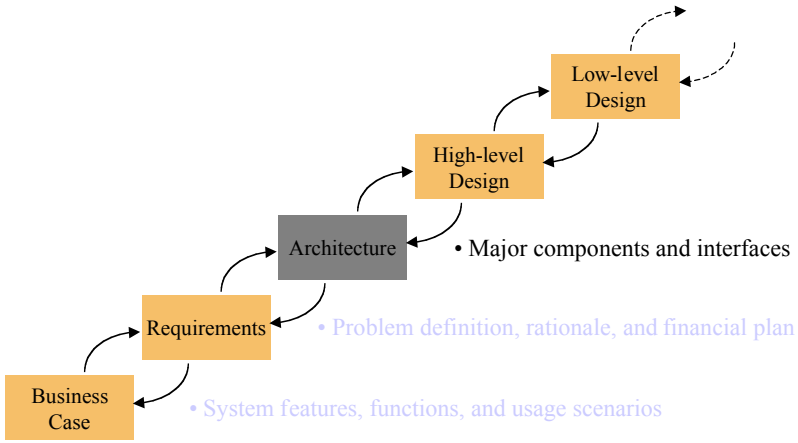
The Role of Architecture



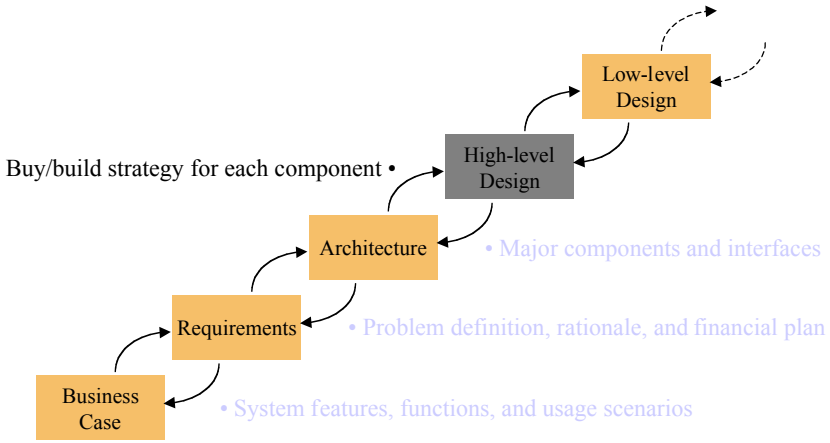
• Problem definition, rationale, and financial plan

• System features, functions, and usage scenarios

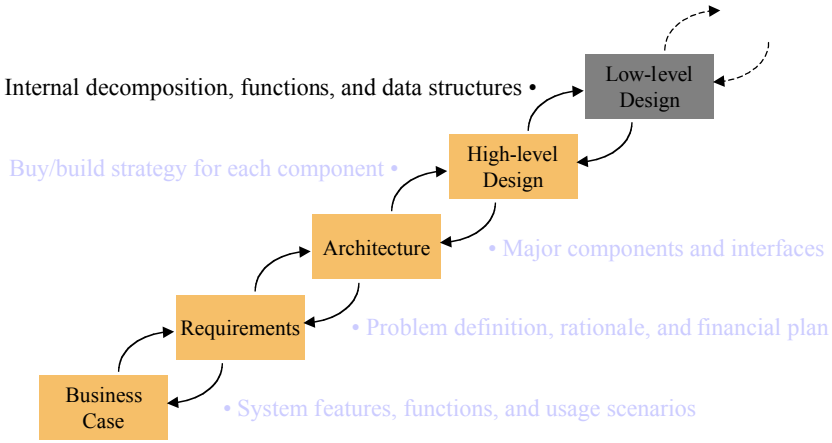
The Role of Architecture



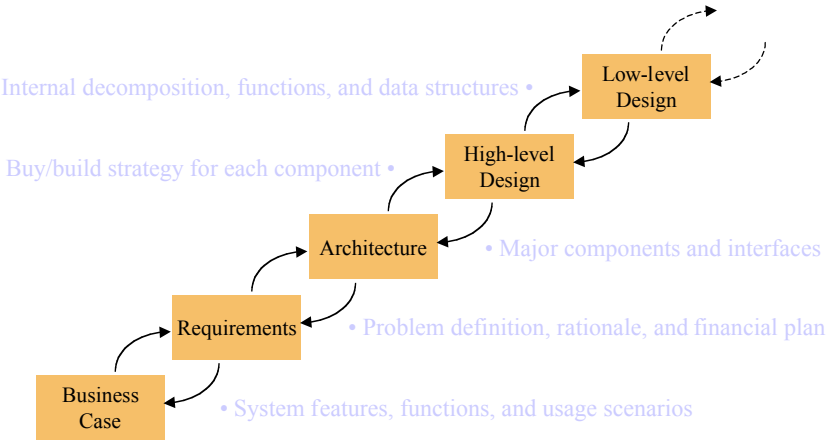
The Role of Architecture



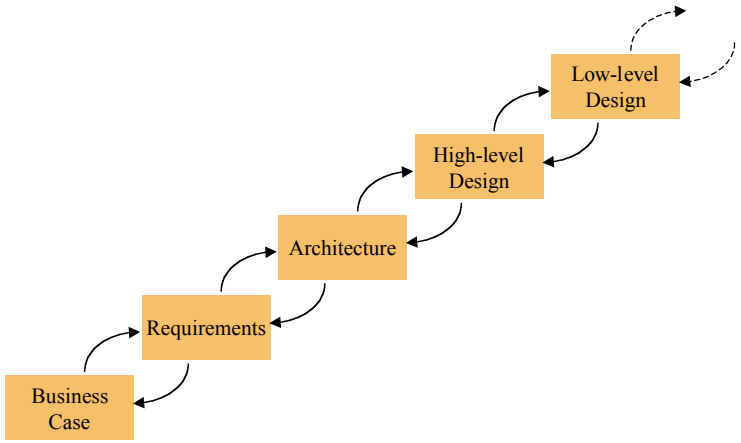
The Role of Architecture



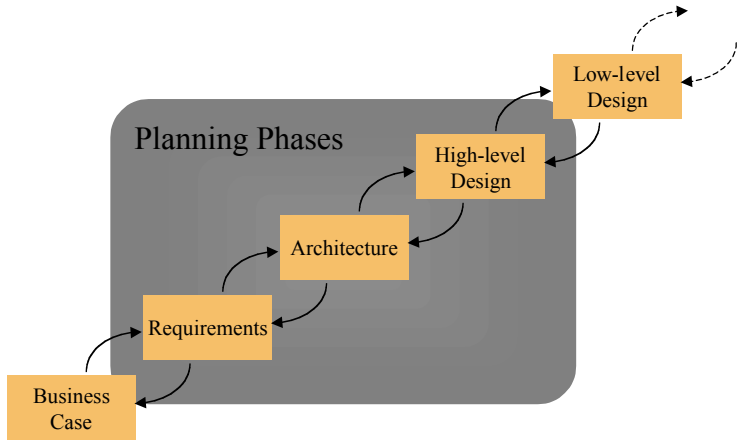
The Role of Architecture



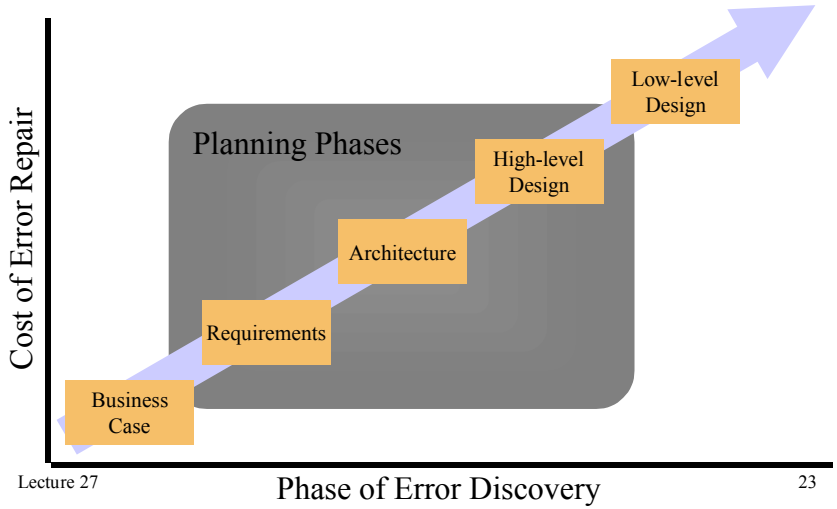
The Central Planning Phase



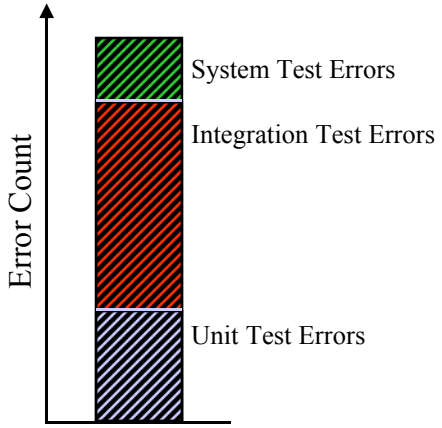
The Central Planning Phase



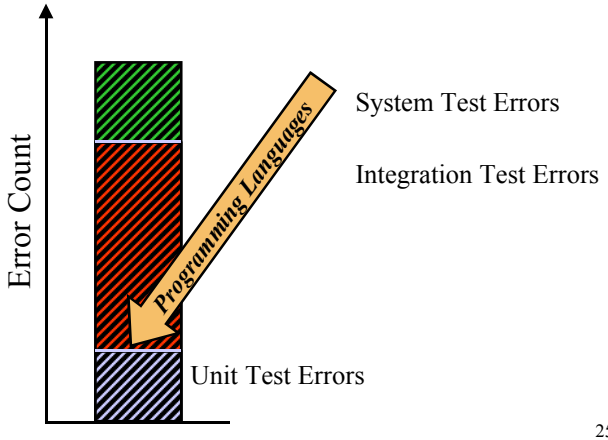
Good Architecture Lowers Cost



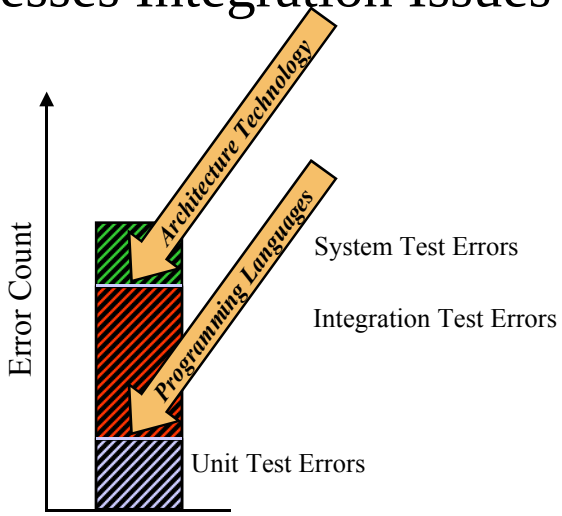
Addresses Integration Issues



Addresses Integration Issues



Addresses Integration Issues



Architectural Domains

Computer Science Domain

- Bus
- Pipes and Filters
- Layered Abstract Machines
- Shared Repository
- Clients and Servers

Architectural Domains

Application Domain

Computer Science Domain

- Bus
- Pipes and Filters
- Layered Abstract Machines
- Shared Repository
- Clients and Servers

Architectural Domains

Application Domain

- Translation
- Data Processing
- Transaction Processing
- Telephone Call Processing
- Flight Dynamics Control

Computer Science Domain

- Bus
- Pipes and Filters
- Layered Abstract Machines
- Shared Repository
- Clients and Servers

Problems ↔ Solutions

Application Domain

Computer Science Domain

Problems ↔ Solutions

Application Domain

A program is compiled by successive application of lexical analysis, syntactic analysis, semantic analysis, and code generation.

Computer Science Domain

Problems ↔ Solutions

Application Domain

A program is compiled by successive application of lexical analysis, syntactic analysis, semantic analysis, and code generation.

Computer Science Domain

The output of one component is the input to another component.

Problems ↔ Solutions

Application Domain

Computer Science Domain

Problems ↔ Solutions

Application Domain

Each agent in the telephone network is a separate computational entity.

Computer Science Domain

Problems ↔ Solutions

Application Domain

Each agent in the telephone network is a separate computational entity.

Computer Science Domain

Each operating system process is a separate computational entity.

Problems ↔ Solutions

Application Domain

Computer Science Domain

Problems ↔ Solutions

Application Domain

Network services are organized as layers, where each layer adds value to services of lower layers. Services at a layer are defined independently of how they are performed.

Computer Science Domain

Problems ↔ Solutions

Application Domain

Network services are organized as layers, where each layer adds value to services of lower layers. Services at a layer are defined independently of how they are performed.

Computer Science Domain

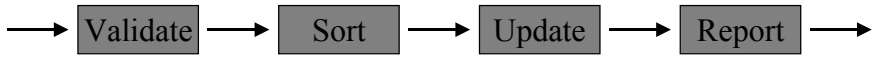
Components are organized as layers, such that each layer provides services only to next higher layer and uses services only of next lower layer.

Architectural Style

- Generic set of components and arrangement of those components
- Constraint on components and their interconnections
- Often given a name...convenient for quickly conveying essential information

Example: DBMS Architecture

- Traditional Business Data Processing
  - Processing steps are independent programs
  - Each runs to completion before next step starts
  - Data stored and passed through magnetic tapes



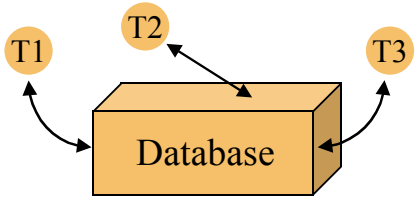
Example: DBMS Architecture

- Traditional Business Data Processing
  - Processing steps are independent programs
  - Each runs to completion before next step starts
  - Data stored and passed through magnetic tapes

Batch Sequential Architecture

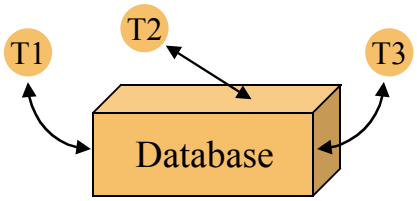
Example: DBMS Architecture

- Centralized Database
  - Monolithic query/update/store engine
  - Applications as serializable transactions



Example: DBMS Architecture

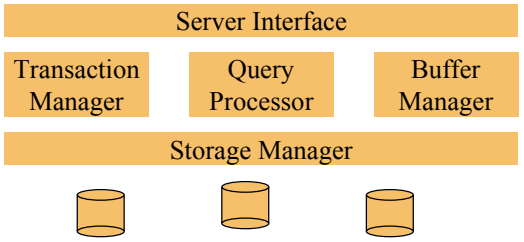
- Centralized Database
  - Monolithic query/update/store engine
  - Applications as serializable transactions



Shared Repository Architecture

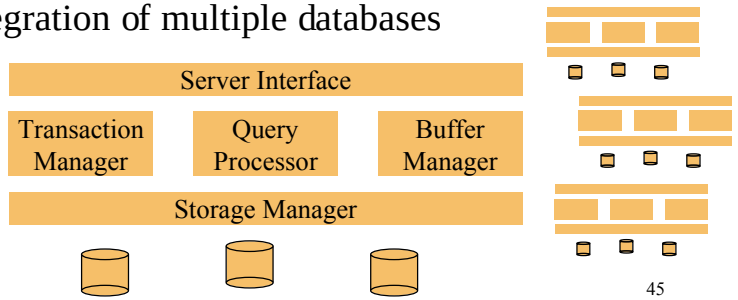
Example: DBMS Architecture

- Modern Database Toolkits
  - Discrete, interchangeable components
  - Applications as distributed clients



Example: DBMS Architecture

- Modern Database Toolkits
  - Discrete, interchangeable components
  - Applications as distributed clients
  - Integration of multiple databases



Analogy: Chemical Engineering

Chemical engineering evolved from a mixture of craft, mysticism, wrong theories, and empirical guesses… Improvements were very slow until the Scientific Revolution…Only then were mystical interpretations replaced by scientific theories: though the early theories were often wrong, they…played a leading role in stimulating thought.

- J.T. Davies

Analogy: Chemical Engineering

- Interesting Architectural Points
  - Theory ignored by engineers
  - "Hacking" worked until problems with scale
  - Scale problems solved by development of relatively small number of *unit operations*
  - Strong emphasis on relationship to process
  - Handbook of chemical engineering eventually developed and used

Unit Operations of Chemical Engineering

- Distillation
- Evaporation
- Drying
- Filtration
- Absorption
- Extraction

Unit Operations of Software Engineering

- Input Validation
- Status Monitoring
- Load Balancing
- Translation
- Filtering
- Multicasting
- …

Analogy: Computer Engineering

- Interesting Architectural Points
  - Relatively small number of component kinds
  - Properties constrained by physical laws
  - Scale by replicating components
- Compared to Software Architecture
  - Very large number of component kinds
  - Constraints hard to determine
  - Scale by adding new component kinds

Analogy: Civil Engineering

- Architectural Styles

  Colonial, Victorian, Ranch, etc.
- Building Codes

  Electrical, structural, zoning, etc.
- Special Expertise

  Slate roofs, post and beam, logs, etc.

Analogy: Civil Engineering

- Architectural Styles

  Colonial, Victorian, Ranch, etc.
  - *Pipes and filters, layers, client/server, etc.*
- Building Codes

  Electrical, structural, zoning, etc.
- Special Expertise

  Slate roofs, post and beam, logs, etc.

Analogy: Civil Engineering

- Architectural Styles

  Colonial, Victorian, Ranch, etc.

  ⧫ *Pipes and filters, layers, client/server, etc.*
- Building Codes

  Electrical, structural, zoning, etc.

  ⧫ *Formal specifications*
- Special Expertise

  Slate roofs, post and beam, logs, etc.

Analogy: Civil Engineering

- Architectural Styles

  Colonial, Victorian, Ranch, etc.

  ⧫ *Pipes and filters, layers, client/server, etc.*
- Building Codes

  Electrical, structural, zoning, etc.

  ⧫ *Formal specifications*
- Special Expertise

  Slate roofs, post and beam, logs, etc.

  ⧫ *Domain-specific architectures*