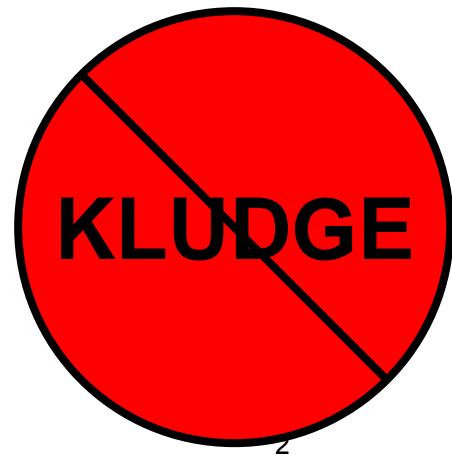# SE 4352
Software Architecture and Design

Fall 2018

Module 2

KLUDGE

1. A system, especially a computer system, that is constituted of poorly matched elements or of elements originally intended for other applications

2. A clumsy or inelegant solution to a problem

Definition From Websters Dictionary

Software Architecture

- *Software architecture* is process of designing the global organization of a software system, including:
  - ☐ Dividing software into subsystems.
  - ☐ Deciding how these will interact.
  - ☐ Determining their interfaces.
    - The architecture is the core of the design, so all software engineers need to understand it.
    - The architecture will often constrain the overall efficiency, reusability and maintainability of the system.

Chapter 1

Software Architecure in Practice
Third Edition

Len Bass · Paul Clements · Rick Kazman
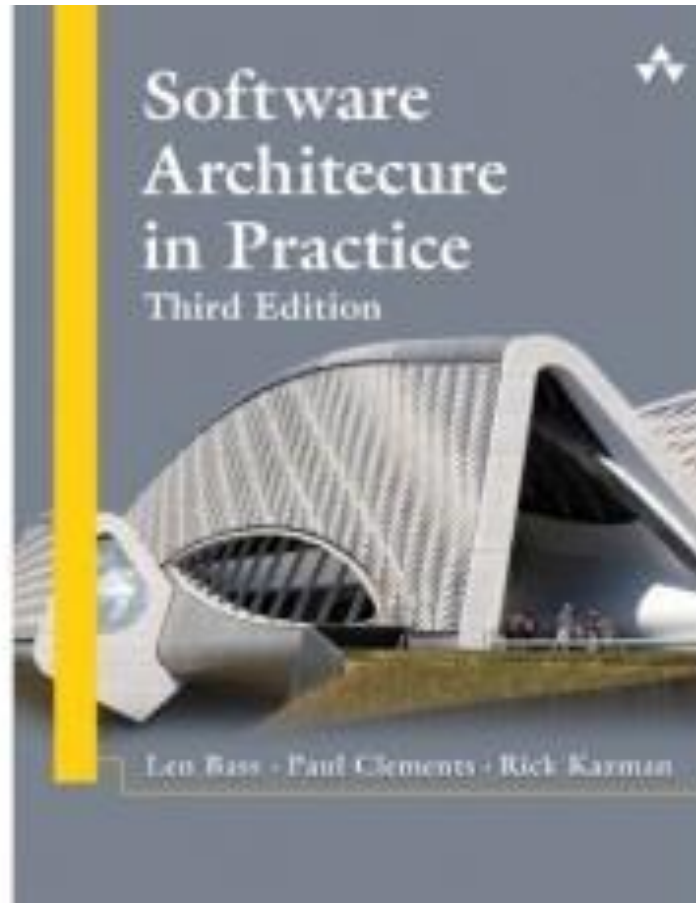
What is Software Architecture?

The software architecture of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both.

Definition

- This definition stands in contrast to other definitions that talk about the system's "early" or "major" design decisions.
  - ☐ Many architectural decisions are made early, but not all are.
  - ☐ Many decisions are made early that are not architectural.
  - ☐ It's hard to look at a decision and tell whether or not it's "major."
- Structures, on the other hand, are fairly easy to identify in software, and they form a powerful tool for system design.

Architecture Is a Set of Software Structures

- A structure is a set of elements held together by a relation.

- Software systems are composed of many structures, and no single structure holds claim to being the architecture.

Which Structures are Architectural?

- A structure is architectural if it supports reasoning about the system and the system's properties.
- The reasoning should be about an attribute of the system that is important to some stakeholder.
- These include
  - □ functionality achieved by the system
  - □ the availability of the system in the face of faults
  - □ the difficulty of making specific changes to the system
  - □ the responsiveness of the system to user requests,
  - □ many others.

Architecture is an Abstraction

- An architecture comprises software elements and how the elements relate to each other.
  - □ specifically omits certain information about that is not useful for reasoning about the system.
  - □ omits information that has no ramifications outside of a single element.
  - □ An architecture selects certain details and suppresses others.
  - □ Private details of elements—details having to do solely with internal implementation—are not architectural.
- The architectural abstraction lets us look at the system in terms of its elements, how they are arranged, how they interact, how they are composed, what their properties are that support our system reasoning, and so forth.
- This abstraction is essential to taming the complexity of an architecture.
- We simply cannot, and do not want to, deal with all of the complexity all of the time.

Every System has a Software Architecture

- Every system comprises elements and relations among them to support some type of reasoning.
- But the architecture may not be known to anyone.
  - Perhaps all of the people who designed the system are long gone
  - Perhaps the documentation has vanished (or was never produced)
  - Perhaps the source code has been lost (or was never delivered)
- An architecture can exist independently of its description or specification.
- Documentation is critical.

Architecture Includes Behavior

- The behavior of each element is part of the architecture insofar as that behavior can be used to reason about the system.
- This behavior embodies how elements interact with each other, which is clearly part of the definition of architecture.
- Box-and-line drawings that are passed off as architectures are not architectures at all.
  - When looking at the names of the boxes a reader may well imagine the functionality and behavior of the corresponding elements.
  - But it relies on information that is not present – and could be wrong!
- This does not mean that the exact behavior and performance of every element must be documented in all circumstances.
  - Some aspects of behavior are fine-grained and below the architect's level of concern.
- To the extent that an element's behavior influences another element or influences the acceptability of the system as a whole, this behavior must be considered, and should be documented, as part of the software architecture.

Physiological Structures
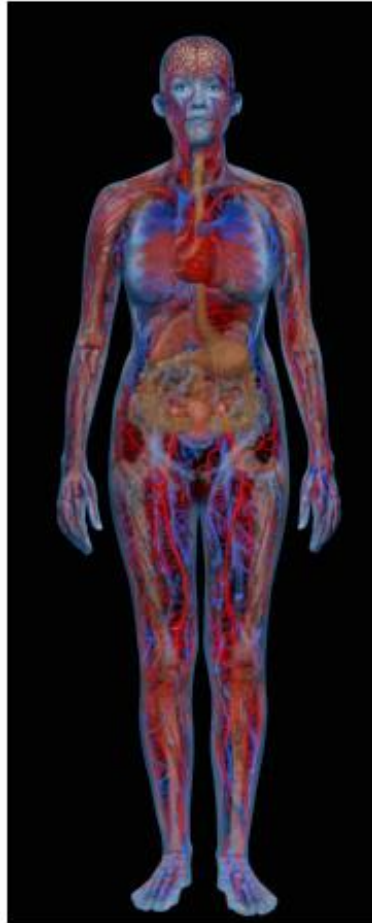
Physiological Structures

- The neurologist, the orthopedist, the hematologist, and the dermatologist all have different views of the structure of a human body.
- Ophthalmologists, cardiologists, and podiatrists concentrate on specific subsystems.
- The kinesiologist and psychiatrist are concerned with different aspects of the entire arrangement's behavior.
- Although these views are pictured differently and have different properties, all are inherently related, interconnected.
- Together they describe the architecture of the human body.
- So it is with software!

Structures and Views

- A *view* is a representation of a coherent set of architectural elements, as written by and read by system stakeholders.
  - A view consists of a representation of a set of elements and the relations among them.
- A *structure* is the set of elements itself, as they exist in software or hardware.
- In short, a view is a representation of a structure.
  - For example, a module *structure* is the set of the system's modules and their organization.
  - A module *view* is the representation of that structure, documented according to a template in a chosen notation, and used by some system stakeholders.
- Architects design structures. They document views of those structures.

Structures Provide Insight

- Structures play such an important role in our perspective on software architecture because of the analytical and engineering power they hold.
- Each structure provides a perspective for reasoning about some of the relevant quality attributes.
- For example:
  - The module structure, which embodies what modules use what other modules, is strongly tied to the ease with which a system can be extended or contracted.
  - The concurrency structure, which embodies parallelism within the system, is strongly tied to the ease with which a system can be made free of deadlock and performance bottlenecks.
  - The deployment structure is strongly tied to the achievement of performance, availability, and security goals.
  - And so forth.

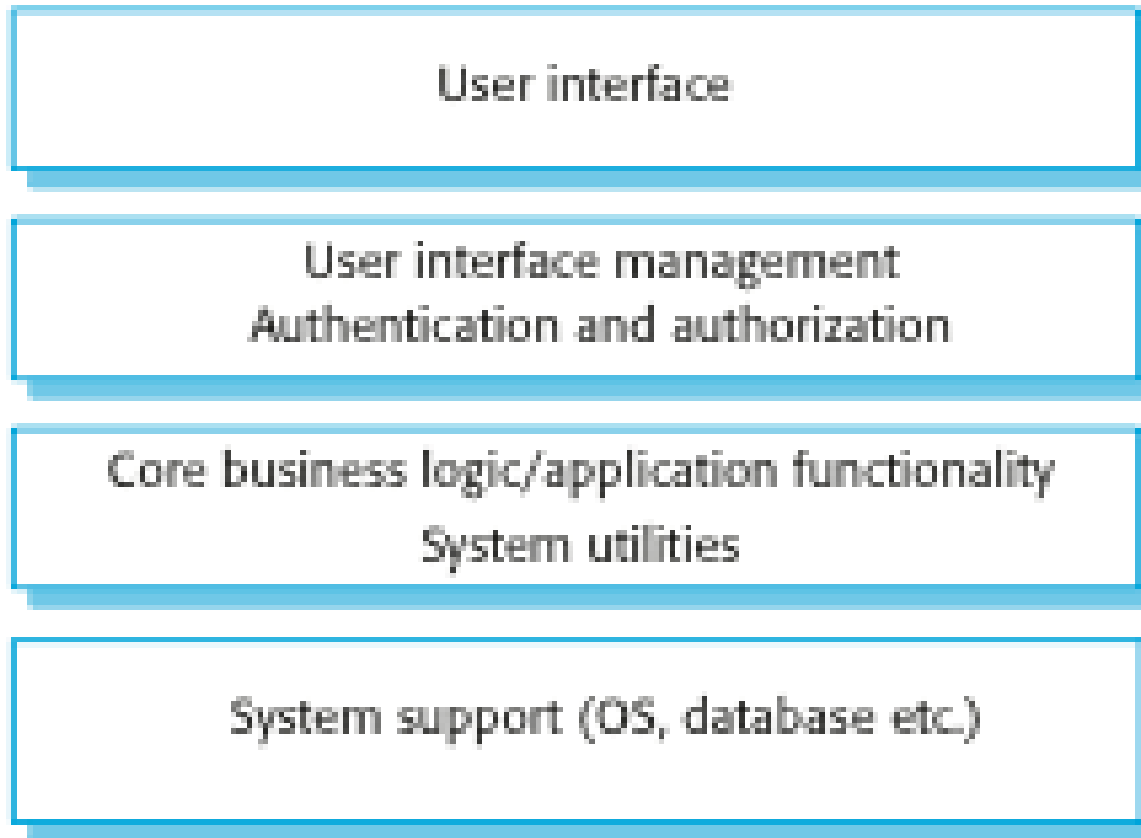# Relating Structures to Each Other

- Elements of one structure will be related to elements of other structures, and we need to reason about these relations.

    - A module in a decomposition structure may be manifested as one, part of one, or several components in one of the component-and-connector structures.

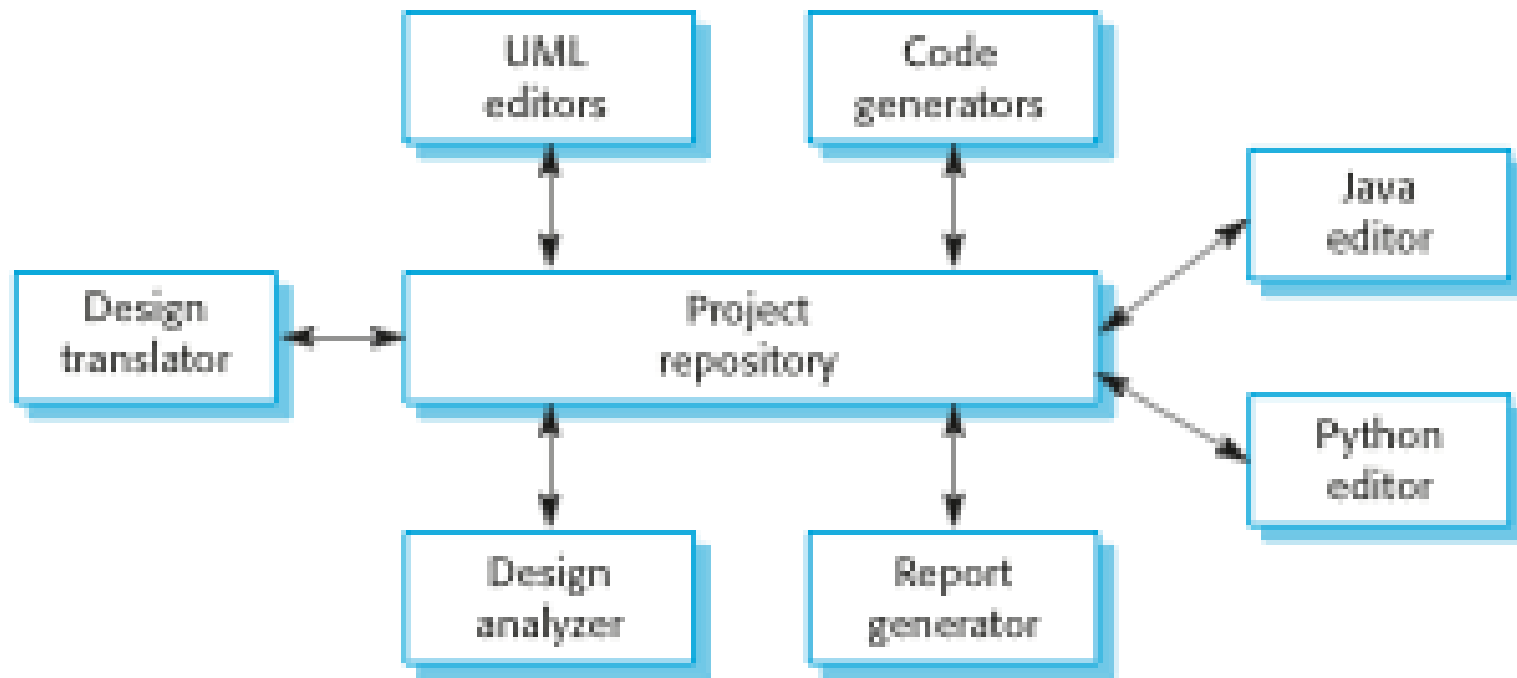- In general, mappings between structures are many to many.

Architectural Patterns

- Architectural elements can be composed in ways that solve particular problems.
  - □ The compositions have been found useful over time, and over many different domains
  - □ They have been documented and disseminated.
  - □ These compositions of architectural elements, called architectural patterns.
  - □ Patterns provide packaged strategies for solving some of the problems facing a system.
- An architectural pattern delineates the element types and their forms of interaction used in solving the problem.
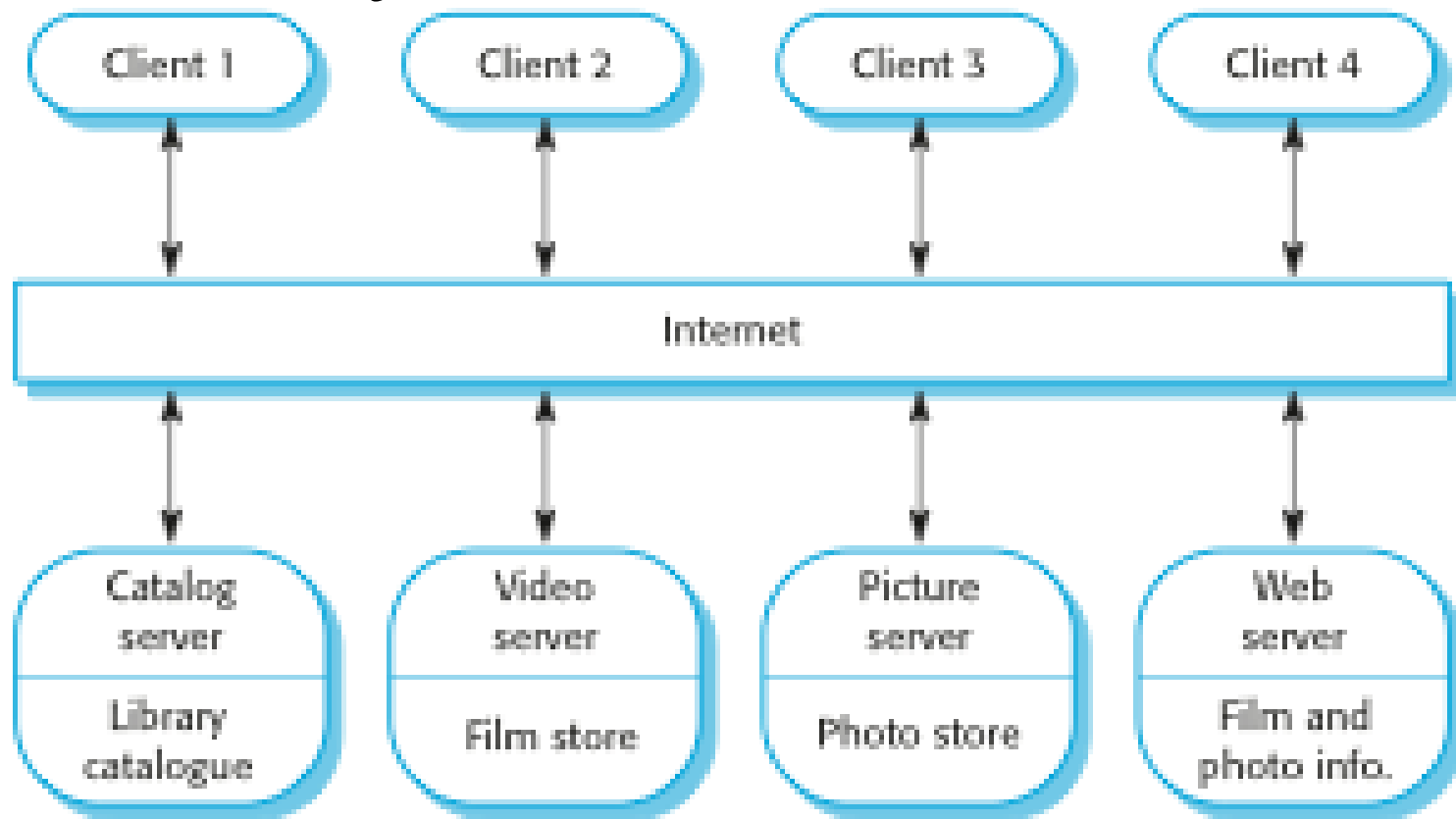
A generic layered architecture

User interface

User interface management Authentication and authorization

Core business logic/application functionality System utilities

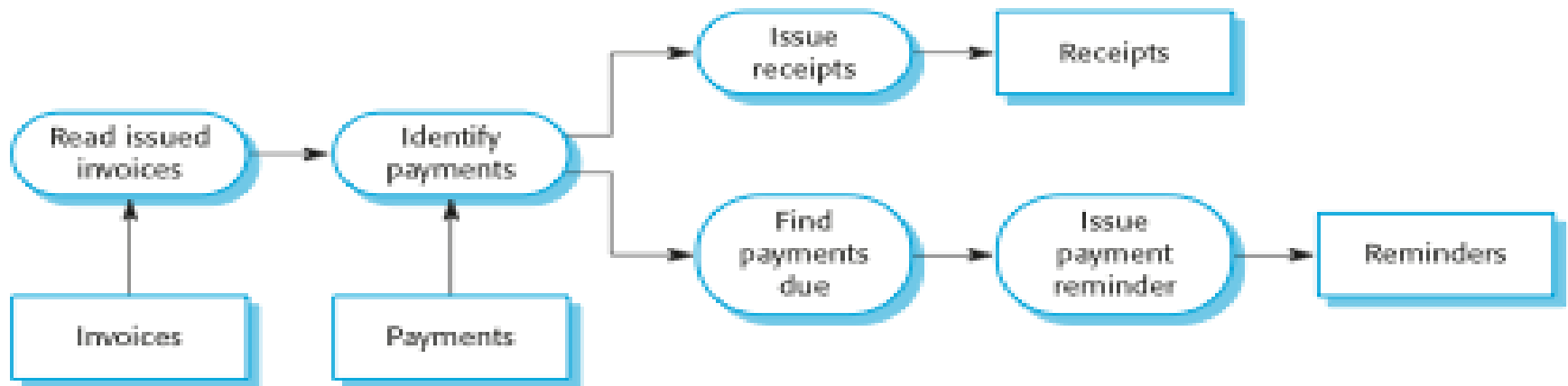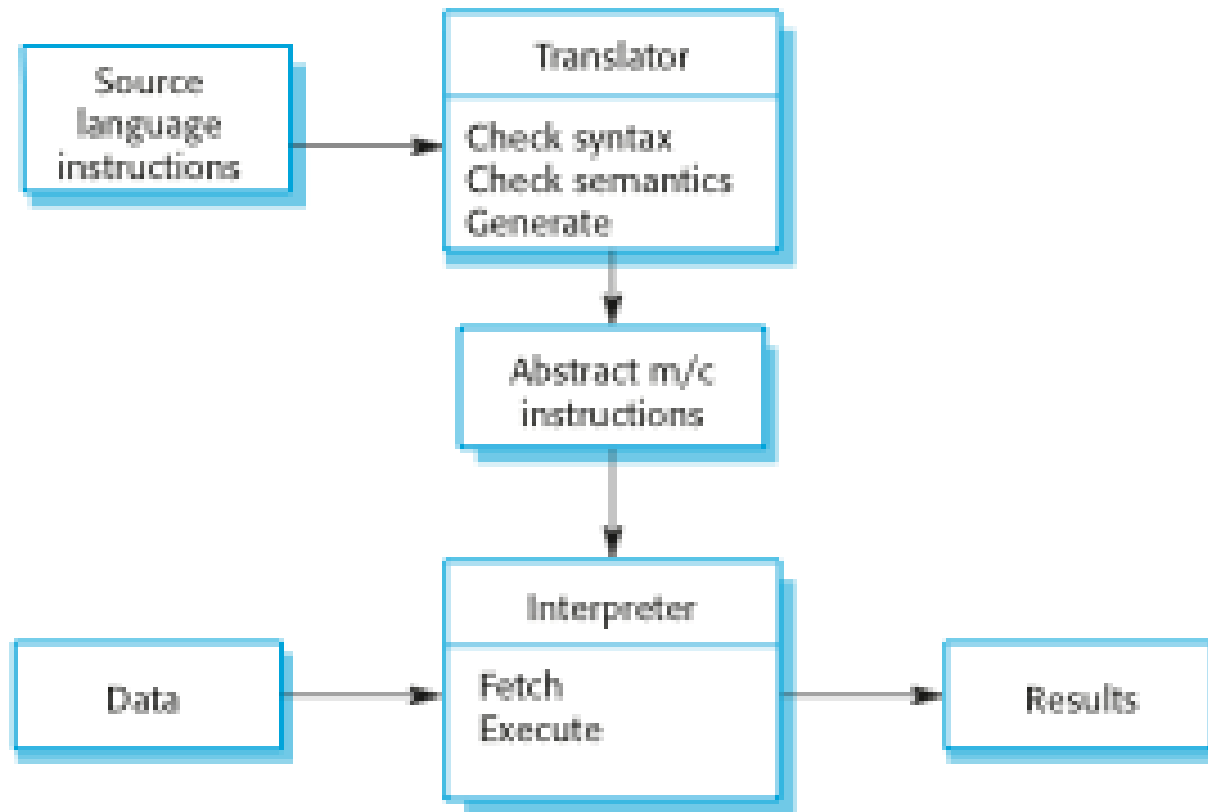| System support (OS, database etc.) |
| --- |

A repository architecture for an IDE

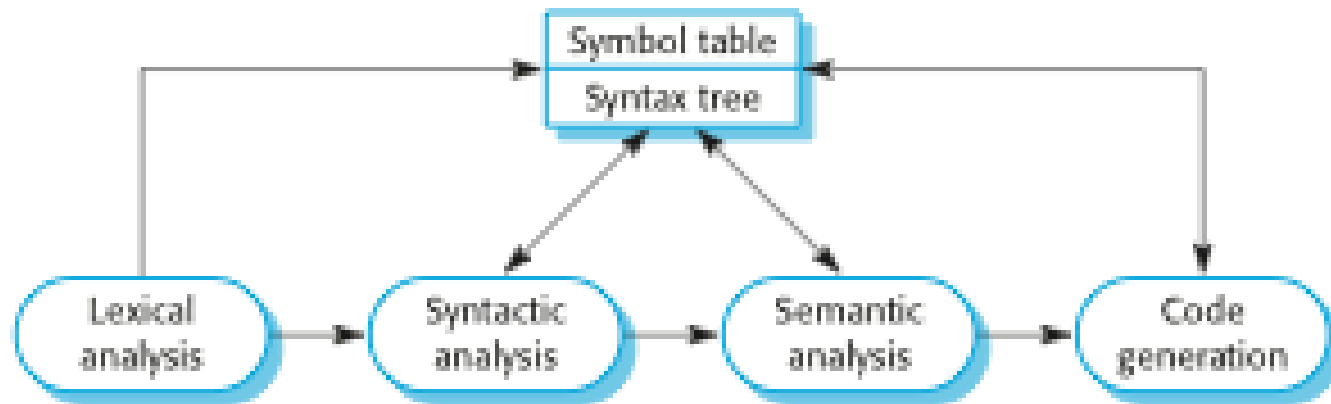# A client–server architecture for a film library

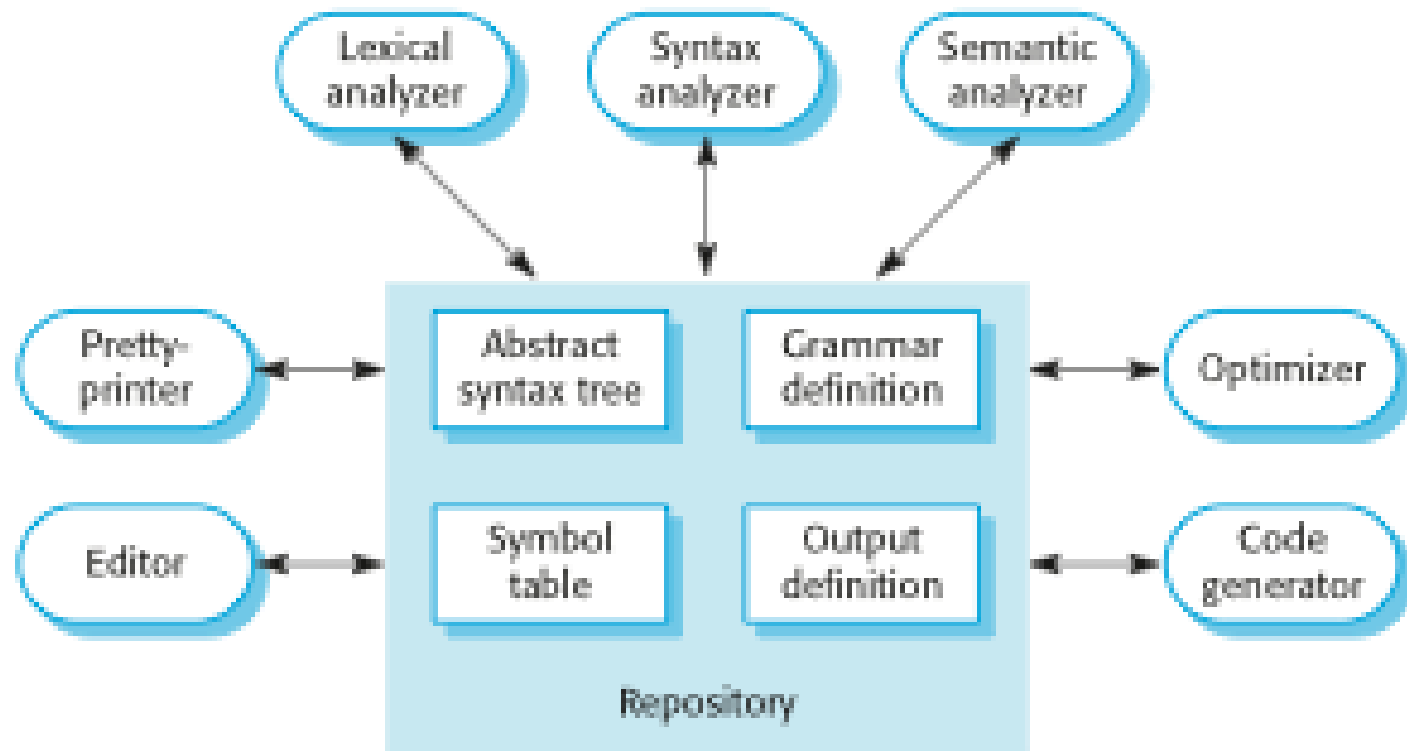# An example of the pipe and filter architecture

The architecture of a language processing system

A pipe and filter compiler architecture

A repository architecture for a language processing system

What Makes a "Good" Architecture?

- There is no such thing as an inherently good or bad architecture.

- Architectures are either more or less fit for some purpose

- Architectures can be evaluated but only in the context of specific stated goals.

- There are, however, good rules of thumb.

Process "Rules of Thumb"

- The architecture should be the product of a single architect or a small group of architects with an identified technical leader.
    - This approach gives the architecture its conceptual integrity and technical consistency.
    - This recommendation holds for Agile and open source projects as well as "traditional" ones.
    - There should be a strong connection between the architect(s) and the development team.
- The architect (or architecture team) should base the architecture on a prioritized list of well-specified quality attribute requirements.
- The architecture should be documented using views. The views should address the concerns of the most important stakeholders in support of the project timeline.
- The architecture should be evaluated for its ability to deliver the system's important quality attributes.
    - This should occur early in the life cycle and repeated as appropriate.
- The architecture should lend itself to incremental implementation,
    - Create a "skeletal" system in which the communication paths are exercised but which at first has minimal functionality.

InClass Group Project

- Teams of 4 to 5
- What is the architecture of an ATM system?
- Write down team answer
- Turn in for display to all students

# The software architecture of an ATM system
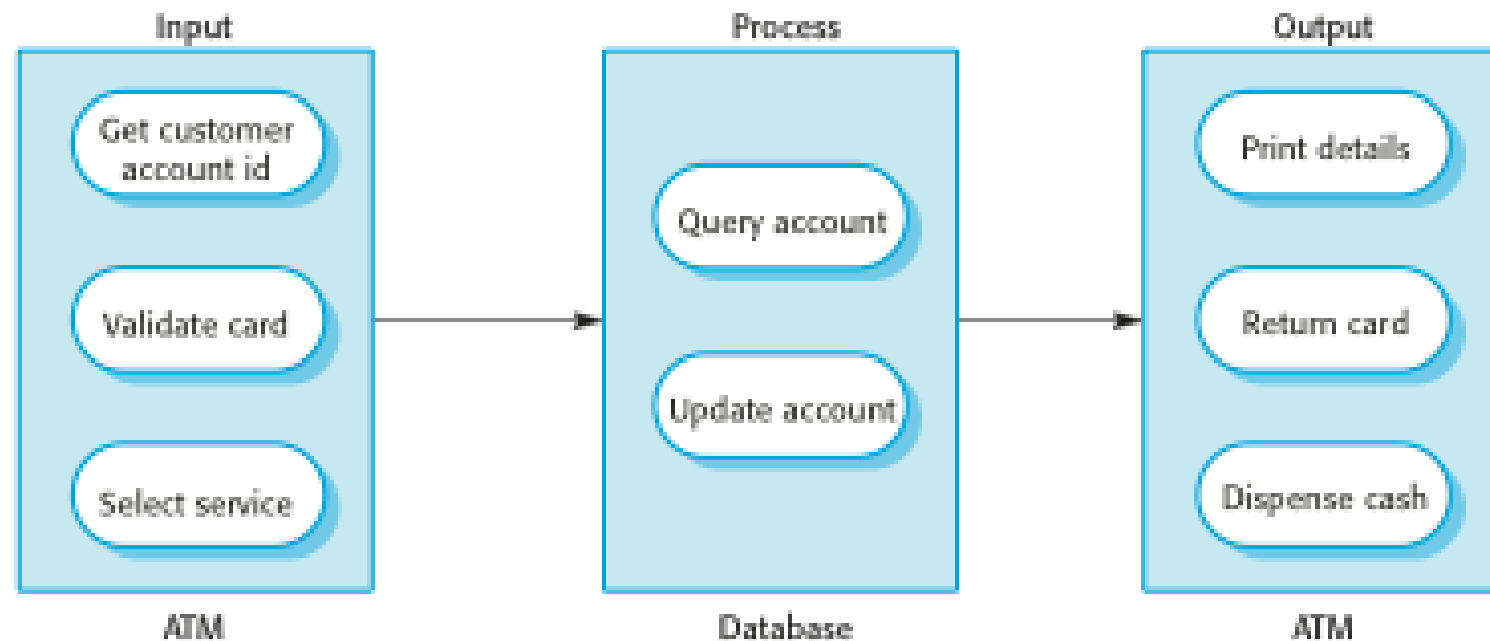
Summary

- The software architecture of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both.

- A structure is a set of elements and the relations among them.

- A view is a representation of a coherent set of architectural elements. A view is a representation of one or more structures.

Summary

- Structures represent the primary engineering leverage points of an architecture.

- Every system has a software architecture, but this architecture may be documented and disseminated, or it may not be.

- There is no such thing as an inherently good or bad architecture. Architectures are either more or less fit for some purpose.

Chapter 3
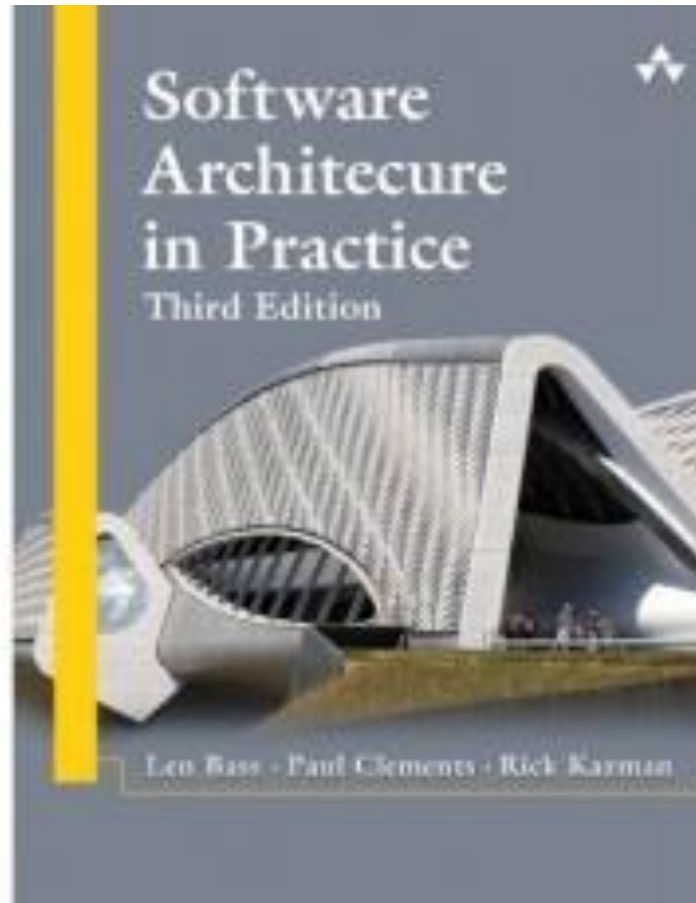
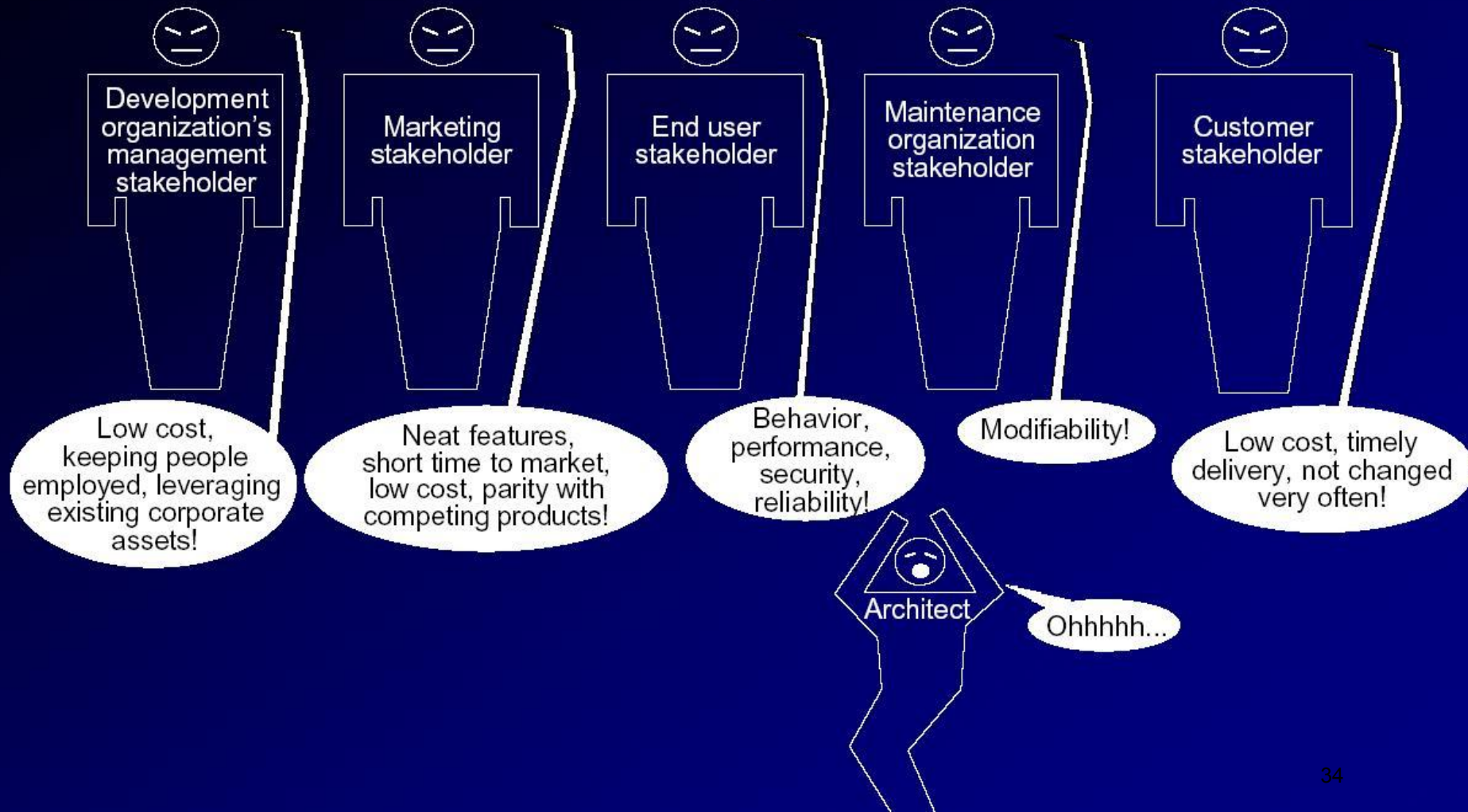# Who is involved in Software Architecture?

One common categorization

- Programming in the small -- easy
  - Simple standalone
- Programming in the large -- not so easy
  - Lots of people and lots of components

Who influences Software Architects?

What does an Architect do?

- What are the responsibilities of the architect?

What does an Architect do?

- What are the responsibilities of the architect?
- Many responsibilities:
  - ☐ Liaison with stakeholders
  - ☐ Technology knowledge
  - ☐ Software engineering
  - ☐ Risk management
- Long list at:
  - ☐ http://www.sei.cmu.edu/ata/arch_duties.html

A Software Architecture Process

- Architects must be versatile:
  - □ **Work with the requirements team:** The architect plays an important role in requirements gathering by understanding the overall systems needs and ensuring that the appropriate quality attributes are explicit and understood.
  - □ **Work with various application stakeholders:** Architects play a pivotal liaison role by making sure all the application's stakeholder needs are understood and incorporated into the design.
  - □ **Lead the technical design team:** Defining the application architecture is a design activity.
  - □ **Work with the project management:** Planning, estimates, budgets, schedules

What does an Architect do?

- What are some of the skillsets that an Architect should have?

What's the goal?

generate value for the customer

Programming & Construction

Architecture
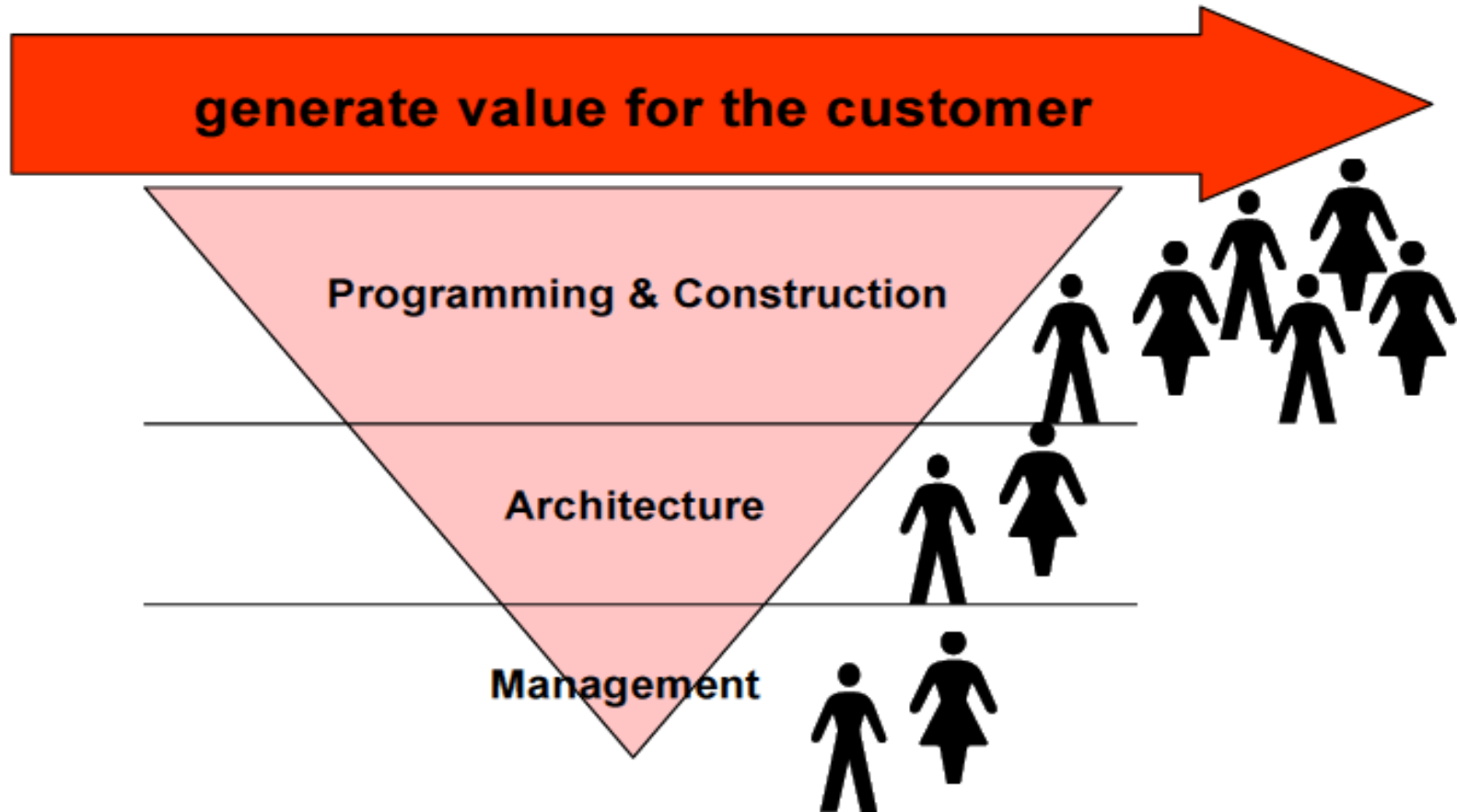
Management

Division of Labor

- Programming Level (Technical skills)
  - ☐ Component implementation
  - ☐ Component customization
- Software Architect Level (Abstraction & Communication Skills)
  - ☐ Component definition
  - ☐ Protocol definition
- Project Management Level (Business & Social Skills)
  - ☐ Requirements engineering
  - ☐ Cost estimation, resource allocation
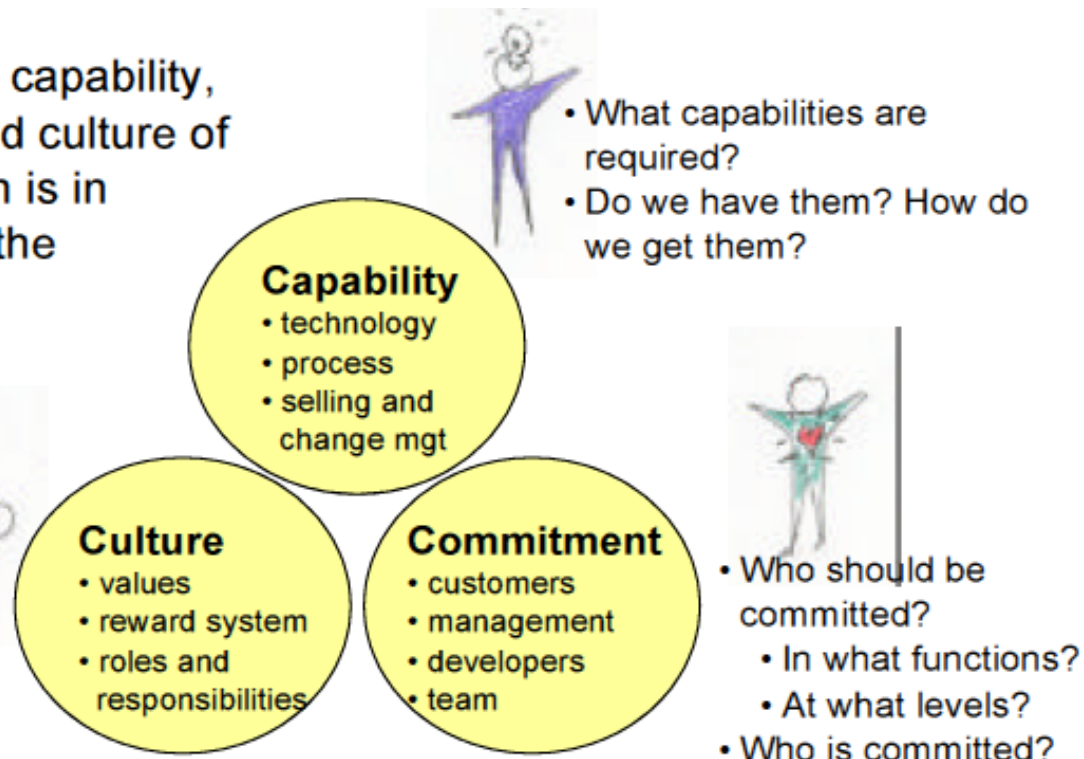 - ☐ Project planning and controlling
  - ☐ Group communication and leaderships

Advice to Managers

Ensure that the capability, commitment and culture of the organization is in alignment with the approach

- What capabilities are required?
- Do we have them? How do we get them?

Capability
- technology
- process
- selling and change mgt

Does the culture support or counter the approach?

Culture
- values
- reward system
- roles and responsibilities

Commitment
- customers
- management
- developers
- team

- Who should be committed?
  - In what functions?
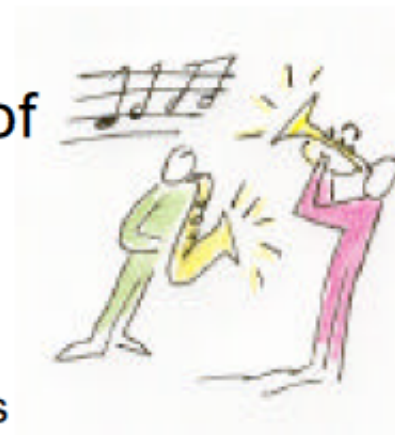 - At what levels?
- Who is committed?

Advice to Architects

- The architecture must serve its stakeholders
  - it *must* address a strategic business objective of the key sponsor(s)
  - it *must* contribute immediate value to utilizers (developers, project managers, etc.) of the architecture

Advice to Developers

- It the architecture directly impacts you
  - follow the architecture
    - active acceptance of leadership
    - contribute, collaborate to achieve the goal
- If not, don't get in the way
- Both following and getting out of the way involve
  - goodwill, trust
  - putting your ego aside
  - leadership - your behavior affects others

Influences on the Architect