CSE-C3610 Software Engineering, 5 cr

Software Architecture

Casper Lassenius
12.2.2013

Topics covered

- What is software architecture?
- Why is software architecture important?
- Who does software architecture?
- When should you do software architecture?
- Architectural patterns

Software Architecture: What?

Software Architecture: What?

- "…the **structure** or structures of the system, which comprise software **elements**, the **externally visible properties** of those elements, and the **relationships** among them."

Bass et. allii, 2003

Software Architecture: What?

- "…the set of **significant decisions** about the **organization** of a software system, the **selection** of the **structural elements** and their **interfaces** by which the system is composed, together with their **behavior** as specified in the collaborations among those elements, the **composition** of these structural and behavioral elements into progressively larger **subsystems**, and the **architectural style** that guides this organization---these elements and their interfaces, their collaborations, and their composition"

Kruchten, 1999

Why an Architecture?

- All systems have an architecture—benefits of an explicitly understood architecture:
  - Stakeholder communication
    - Architecture may be used as a focus of discussion by system stakeholders.
  - System analysis
    - Means that analysis of whether the system can meet its non-functional requirements is possible.
  - Large-scale reuse
    - The architecture may be reusable across a range of systems
    - Product-line architectures may be developed.

Architectural Design

- An early stage of the system design process or conducted mainly in the early iterations when doing IID
- Represents the link between specification and design processes
- Often carried out in parallel with some specification activities
- It involves identifying major system components and their communications

Architectural Design Decisions

- Often called architecturally significant decisions
- Architectural design is a creative process so the process differs depending on the type of system being developed.
- However, a number of common decisions span all design processes and these decisions affect the non-functional characteristics of the system.

Architectural design decisions

- Is there a generic application architecture that can be used?
- How will the system be distributed?
- What architectural styles are appropriate?
- What approach will be used to structure the system?
- How will the system be decomposed into modules?
- What control strategy should be used?
- How will the architectural design be evaluated?
- How should the architecture be documented?

Architecture and Non-Functional Attributes

- Performance
  - Localise critical operations and minimize communications. Use large rather than fine-grain components.
- Security
  - Use a layered architecture with critical assets in the inner layers.
- Safety
  - Localize safety-critical features in a small number of sub-systems.
- Availability
  - Include redundant components and mechanisms for fault tolerance.
- Maintainability
  - Use fine-grain, replaceable components.

Architectural Representations

- Simple, informal block diagrams showing entities and relationships are the most frequently used method for documenting software architectures.
- But these have been criticized because they lack semantics, do not show the types of relationships between entities nor the visible properties of entities in the architecture.
- Depends on the use of architectural models.The requirements for model semantics depends on how the models are used.

Architectural Views

- What views or perspectives are useful when designing and documenting a system's architecture?
- What notations should be used for describing architectural models?
- Each architectural model only shows one view or perspective of the system.
  - It might show how a system is decomposed into modules, how the run-time processes interact or the different ways in which system components are distributed across a network. For both design and documentation, you usually need to present multiple views of the software architecture.

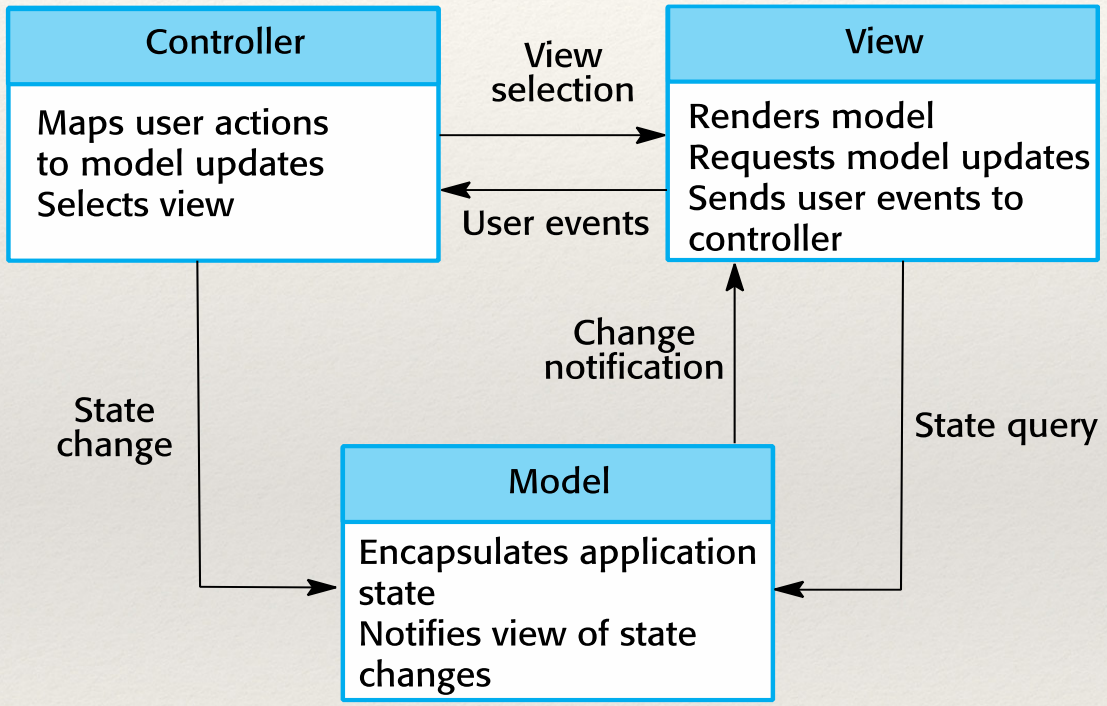
4 + 1 view model of software architecture

- A **logical** view, which shows the key abstractions in the system as objects or object classes.
- A **process** view, which shows how, at run-time, the system is composed of interacting processes.
- A **development** view, which shows how the software is decomposed for development.
- A **physical** view, which shows the system hardware and how software components are distributed across the processors in the system.
- Related using example *use cases* or *scenarios* (+1)

Architectural Patterns

- Patterns are a means of representing, sharing and reusing knowledge.
- An architectural pattern is a stylized description of good design practice, which has been tried and tested in different environments.
- Patterns should include information about when they are and when the are not useful.
- Patterns may be represented using tabular and graphical descriptions.

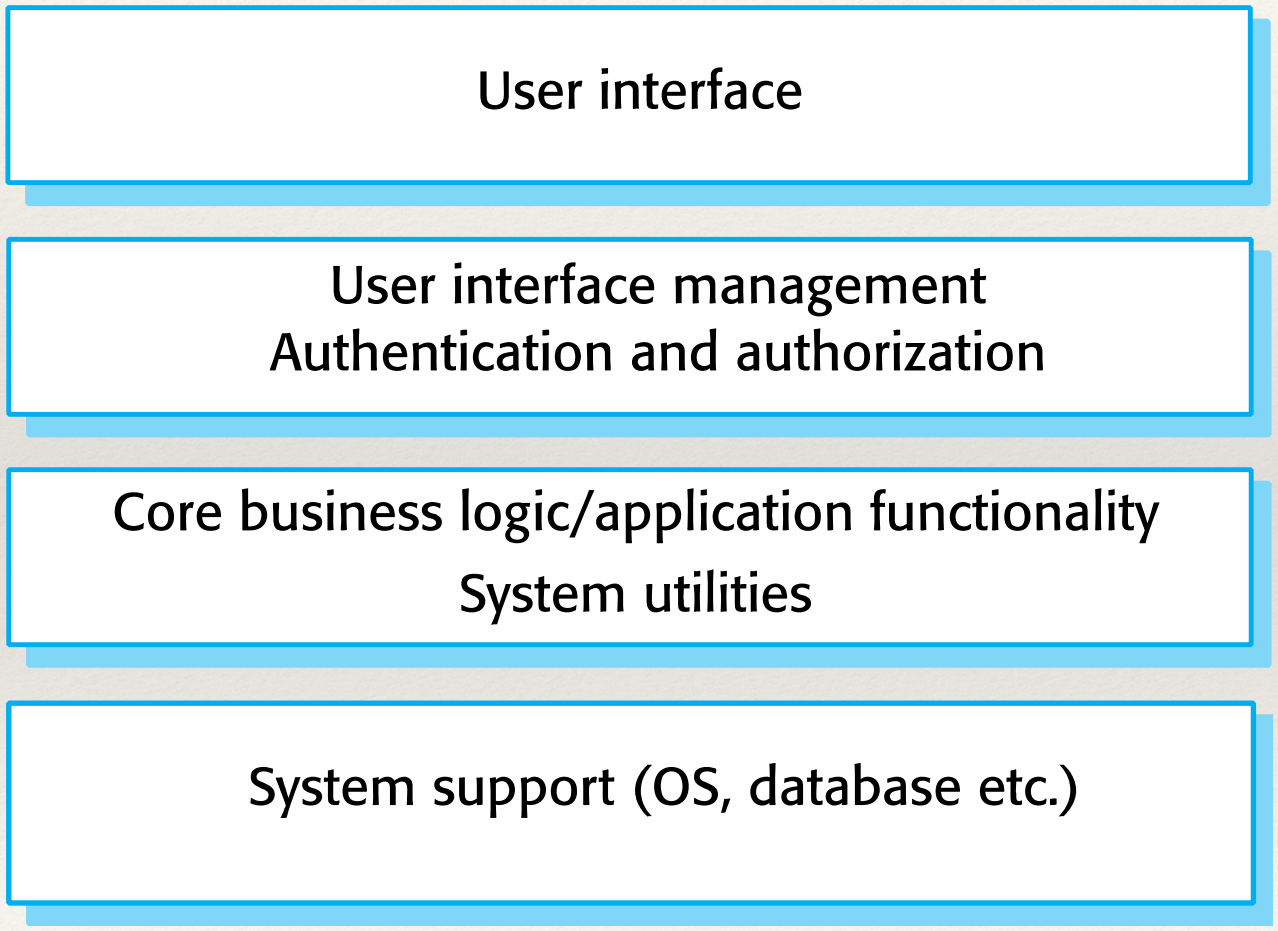
The Model-View-Controller Pattern



Layered Architecture

- Used to model the interfacing of sub-systems.
- Organizes the system into a set of layers (or abstract machines) each of which provide a set of services.
- Supports the incremental development of sub-systems in different layers. When a layer interface changes, only the adjacent layer is affected.
- However, often artificial to structure systems in this way.

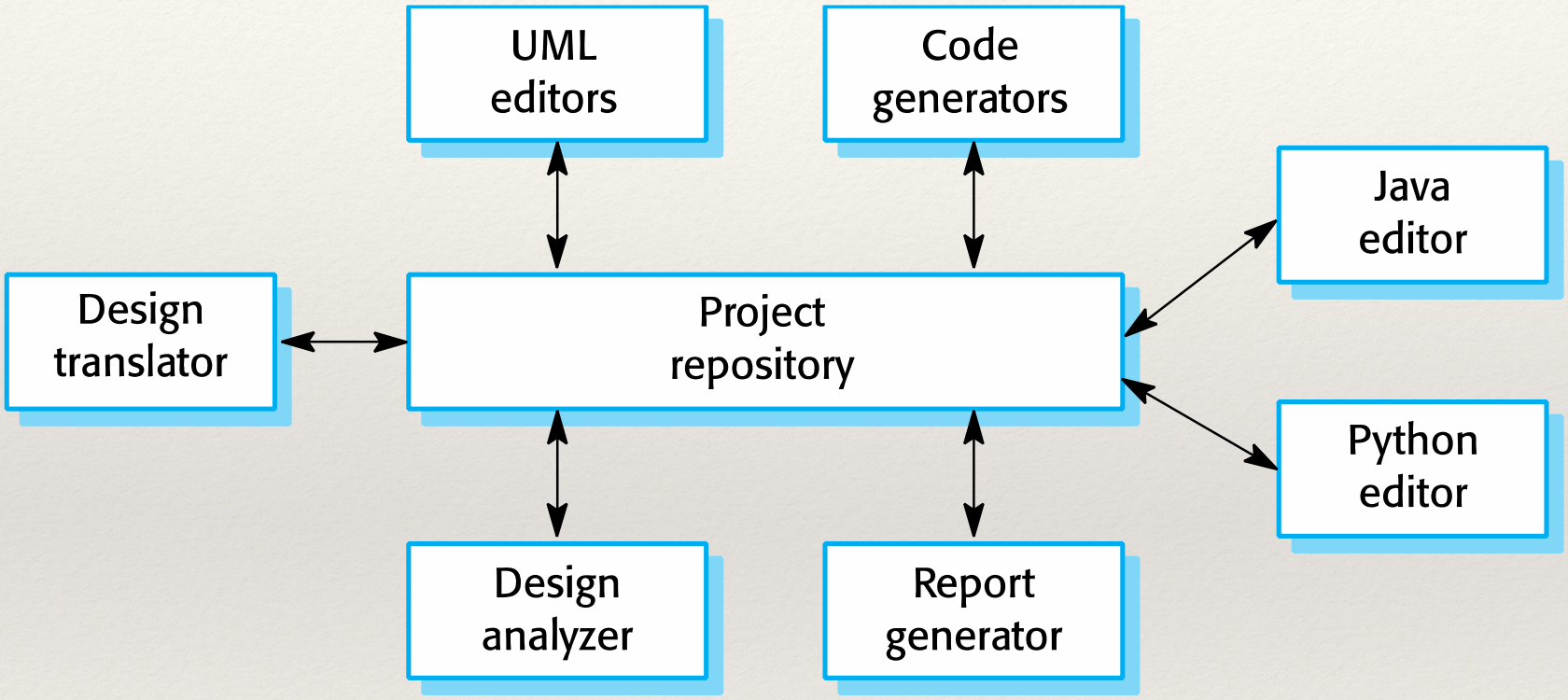
A Generic Layered Architecture



Repository architecture

- Sub-systems must exchange data. This may be done in two ways:
  - Shared data is held in a central database or repository and may be accessed by all sub-systems;
  - Each sub-system maintains its own database and passes data explicitly to other sub-systems.
- When large amounts of data are to be shared, the repository model of sharing is most commonly used a this is an efficient data sharing mechanism.

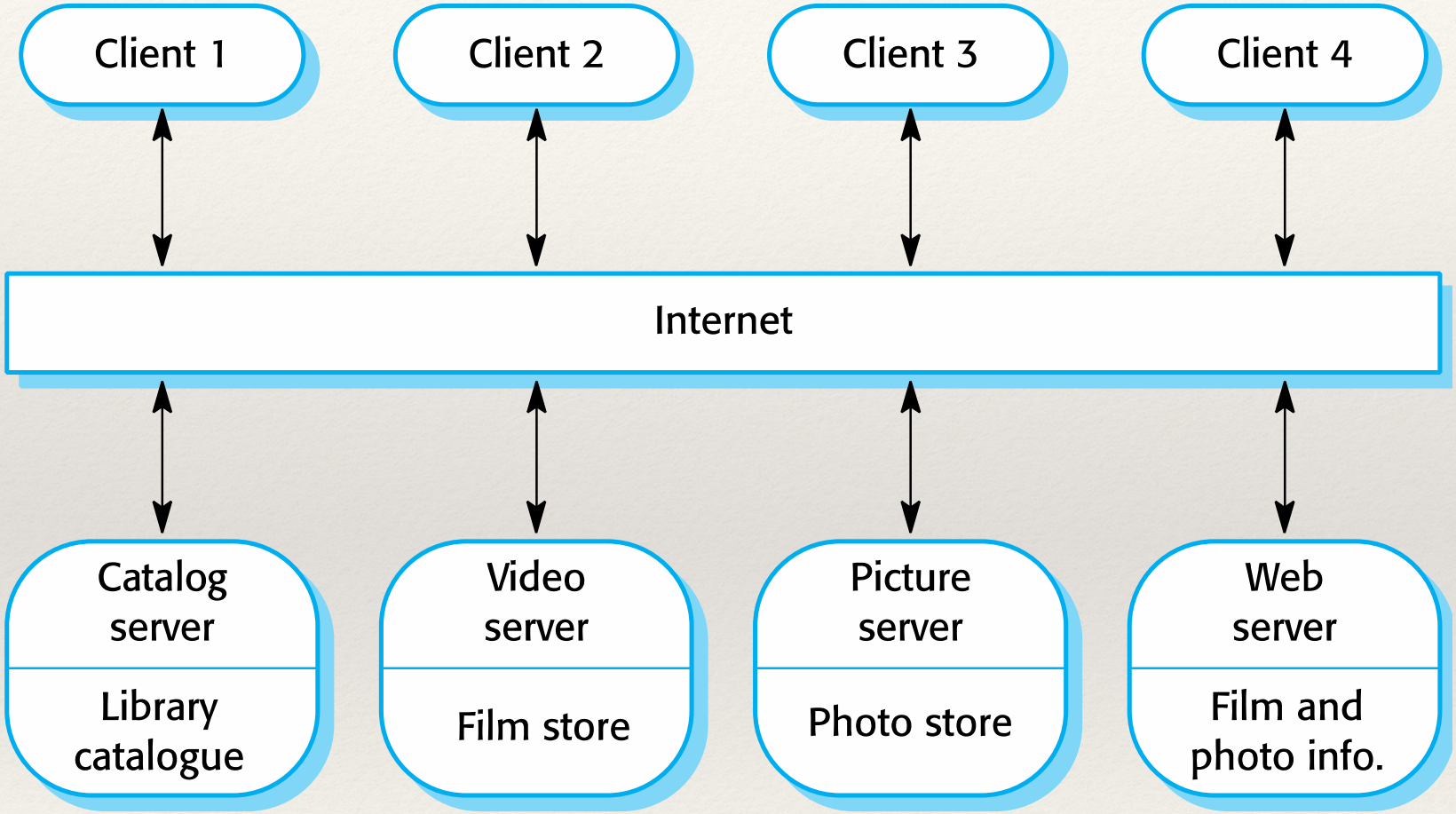
A repository architecture for an IDE



Client-Server Architecture

- Distributed system model which shows how data and processing is distributed across a range of components.
  - Can be implemented on a single computer.
- Set of stand-alone servers which provide specific services such as printing, data management, etc.
- Set of clients which call on these services.
- Network which allows clients to access servers.

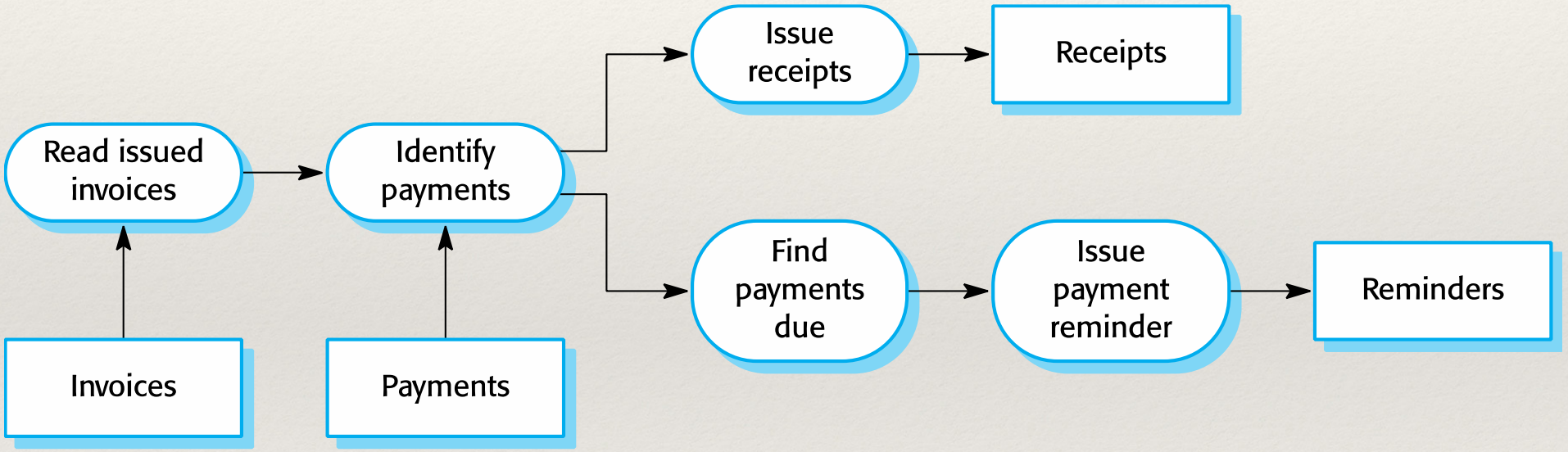
A C-S Architecture for a Film Library



Pipe and Filter Architecture

- Functional transformations process their inputs to produce outputs.
- May be referred to as a pipe and filter model (as in UNIX shell).
- Variants of this approach are very common. When transformations are sequential, this is a batch sequential model which is extensively used in data processing systems.
- Not really suitable for interactive systems.

An Example of the Pipe and Filter Architecture



Summary

- Architecture describes the structure of a system at a high level of abstraction—key components and their relationships
- Architecture can have a significant impact on the “-ilities", i.e. the non-functional or quality attributes of the system
- A good architecture for complex or novel systems do not simply emerge—some planning needs to go into it
- Architectures are often communicated through some kind of documentation
- Architectural patterns describe typical architectural solutions

Questions?