Software Design and Architecture

[Documenting Software Architectures] – Chapter 09

Lecture Outlines

- Why Document the architecture
- Documenting a view
- Documenting beyond the view
- Documenting Quality

Documenting the architecture

- A software system's architecture may be its most crucial determinant of success or failure.
- Without an adequate architecture that delivers required function as well as quality attributes, the project will fail.
- Communicating an architecture to its stakeholders is as important a job as creating it in the first place.

Documenting the architecture

- Even the best architecture will be useless if the people who need it
  - ✓ do not know what it is;
  - ✓ cannot understand it well enough to use, build, or modify it;
  - ✓ misunderstand it and apply it incorrectly.

- Therefore, it is important to document the architecture design properly.

Documenting the architecture

- Documenting a software architecture is a matter of:
  - ✓Choosing a set of relevant views of the architecture,
  - ✓Documenting each of those views, and then
  - ✓Documenting information that applies to more than one view or to the set of views as a whole.
  - ✓In other words:
    documenting software architecture includes *a way to choose the most* ***relevant views****, standard* ***templates*** *for documenting a view and documenting the information* ***beyond views***, and definitions of the templates' content.

Documenting the architecture

- **Three primary uses for architecture documentation:**
  - ✓**Education** -- introducing people to the project
  - ✓**Communication** -- among stakeholders
  - ✓**Analysis** -- assuring quality attributes

Documenting the architecture

- **To start with documentation, you need to (the goal of documentation):**
  - ✓Select common **views** of Software Architecture Design (SAD)
  - ✓Define **stakeholders**
  - ✓Identify stakeholders' **concerns**
  - ✓Define what and how to document (select **templates**)

Possible Architectural View approaches

- **Kruchten's 4+1 views**
  - ✓ **Logical view**: supports behavioral requirements. Key abstractions, which are objects or object classes
  - ✓ **Process view**: addresses concurrency and distribution. Maps threads to objects.
  - ✓ **Development view**: organization of software modules, libraries, subsystems, units of development.
  - ✓ **Physical view**: maps other elements onto processing and communication nodes.
  - ✓ **"Plus one" view**: Maps the other views onto important use cases to show how they work.

Possible Architectural View approaches(cont.)

- **Siemens Four-Views:**
  - ✓Conceptual view
  - ✓Module interconnection view
  - ✓Execution view
  - ✓Code view
- **Herzum & Sims:**
  - ✓Technical architecture
  - ✓Application architecture
  - ✓Project management architecture
  - ✓Functional architecture

Possible Architectural View approaches (cont.)

- **Software Cost Reduction method:**
  - ✓ Module view: shows modules as units of encapsulation; used to isolate changes and achieve modifiability
  - ✓ Process view: shows processes and how they synchronize and communicate at run-time; used to achieve performance
  - ✓ Uses view: shows programs and how they depend on each other; used to achieve incremental development and the ability to quickly field subsets

Possible Architectural View approaches (cont.)

Planning a **view** set requires understanding the needs of the **stakeholders**, how they will use the prepared documentation, and the resources available.

We covered the 4+1 views in this course since it is more commonly used

Define stakeholders

- **Sample stakeholders:**
  - ✓Project Manager
  - ✓Member of Development Team
  - ✓Testers and Integrators
  - ✓Maintainers
  - ✓Product Line Application Builder
  - ✓Customer
  - ✓End User
  - ✓Analyst
  - ✓Infrastructure Support
  - ✓Current and Future Architect

Identify stakeholders' concerns

- Building a table that records the stakeholders' concerns and associated view helps in prioritizing the selected view based on the needs (some stakeholders may have extra weight)

No.	Stakeholder	Concerns to adddress	Suitable view
1.			
2.			
3.			
4.			
5.			

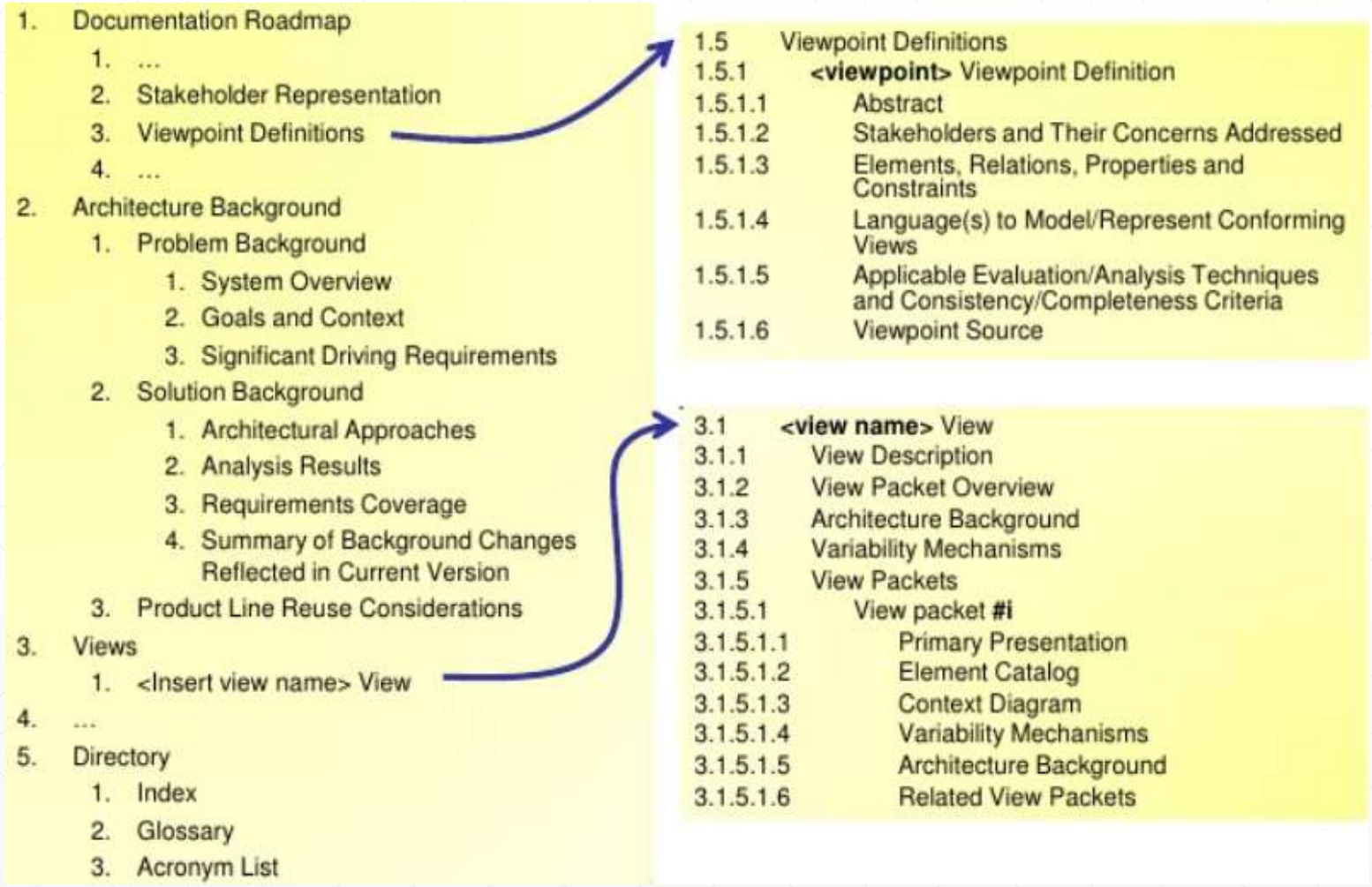
Define How to Document

- After defining identifying the views, stakeholders' concerns, and what is going to be documented, a template is usually selected to prepare the documentation
- The following is an example of the Rational Unified Process template:

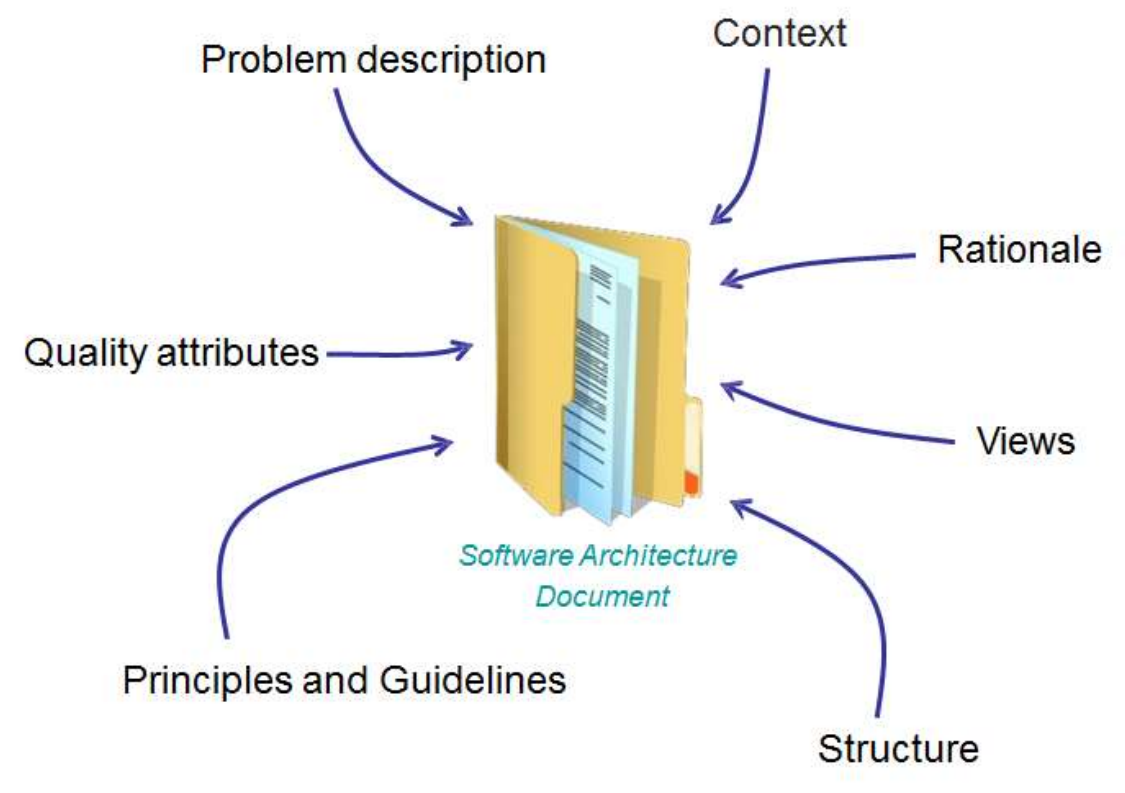
1. Introduction
    1. Purpose
    2. Scope
    3. Definitions, Acronyms, and Abbreviations
    4. References
    5. Overview
2. Architectural Representation
3. Architectural Goals and Constraints
4. Use-Case View
    1. Use-Case Realizations
5. Logical View
    1. Overview
    2. Architecturally Significant Design Packages
6. Process View
7. Deployment View
8. Implementation View
    1. Overview
    2. Layers
9. Data View (optional)
10. Size and Performance
11. Quality

Define How to Document

- The Software Engineering Institute (SEI) template:



Define How to Document



- A software architecture document should have a minimum information necessary to understand the system but no more that that.
- Major necessary information are shown in the figure on the left

Documenting a View

1. **A primary presentation**
   - ✓Usually graphical (we call this a cartoon)
   - ✓May be textual -- e.g., a table
   - ✓If graphical, includes a key explaining the notation (or pointing to explanation)
   - ✓Shows elements and relationships among them
   - ✓Shows information you wish to convey about the view (view packet) first

➢Many times, the primary presentation is all you get. However, it's most likely not enough!

Documenting a View (cont.)

2. An element catalog

- ✓Explains the elements depicted in the primary presentation
- ✓Lists elements and their properties (as defined by the relevant style guide)
- ✓Explains relations, and any exceptions or additions to the relations shown in the primary presentation
- ✓Interfaces of elements

3. A context diagram

- ✓Shows how system (or portion shown in this view packet) relates to its environment.

Documenting a View (cont.)

4. Variability guide:

- ✓ Shows how to exercise any variation points that are a part of the architecture shown in this view.
- ✓ In some architectures, decisions are left unbound until a later stage of the development process, and yet the architecture must still be documented.

5. Architecture background

- ✓ Rationale for design decisions that apply to the entire view (or to that portion of the view being shown), including rejected alternatives and factors that constrained the design
- ✓ Analysis results validating the design decisions
- ✓ Assumptions about the environment and about the need that the system is fulfilling

Documentation beyond views

1. **Documentation roadmap**
   - ✓How the documentation is organized to serve a stakeholder
   - ✓List of views, with the elements/relations of each, and a statement of what the view is for
   - ✓Scenarios for using the documentation, showing which parts should be consulted
2. **View Documentation**
   - ✓Explanation of how each view is documented
   - ✓The standard organization for each view

Documentation beyond views (cont.)

3. **System overview**
   - ✓Short prose description of the system's function, its users, and any important background or constraints.
   - ✓Provides your readers with a consistent mental model of the system and its purpose.
4. **Mapping between views**
   - ✓Establishes useful/insightful correspondence between various views
   - ✓View-to-view associations can be captured as tables

Documentation beyond views (cont.)

5. Rationale:

- ✓ Documents the architectural decisions that apply to more than one view.
  - Documentation of background or organizational constraints or major requirements that led to decisions of system-wide import.
  - Decisions about which fundamental architecture patterns are used.

6. Directory.

- ✓ Set of reference material that helps readers find more information quickly.
  - Index of terms
  - Glossary
  - Acronym list

Documenting Quality Attributes

- The Architecture document should include a rationale that explains the choice of design approach and a discussion about the quality attribute requirements and tradeoffs.
- Architectural elements providing a service often have quality attribute bounds assigned to them
- Architecture documentation often contains a *mapping to requirements* that shows how requirements (including quality attribute requirements) are satisfied.
- Every quality attribute requirement will have a constituency of stakeholders who want to know that it is going to be satisfied. Quality attributes should be clearly identified in the document.

Summary

- You must understand the uses to which the writing is to be put and the audience for the writing.
- Architectural documentation serves as a means for communication among various stakeholders, not only up the management chain and down to the developers but also across to peers.
- An architecture is a complicated artifact, best expressed by focusing on views.
- You must choose the views to document and must choose a set of views that is both minimal and adequate.
- You must document not only the structure of the architecture but also the behavior.

Lecture Outlines

- Why Document the architecture
- Documenting a view
- Documenting beyond the view
- Documenting Quality