

T-76.150 Software Architectures

Documentation

This lecture is based on
Clements et al., Documenting Software Architectures: Views
and Beyond, Addison Wesley, 2002
and
IEEE-1471-2000 Standard



Overview

- Motivation for SA documentation
- Uses of SA documentation
- 7 Rules of sound documentation
- SA documentation package



Architecture design



Software architecture needs to be **designed**

and **documented**



Is documentation *that* important?

- Architecture documentation is important if *communication* of the architecture is important.
 - How can an architecture be used if it cannot be understood?
 - How can an architecture be understood if it cannot be communicated?



Terms

- Specification
 - Formal. Includes capturing rationale?
- Representation
 - Model, abstraction. Gets unnecessarily philosophical?
- Description
 - ADLs. Sounds vague.
- Documentation
 - Connotes creation of an artefact: document

[Clements et al. DSA, 2002]



IEEE 1471 Uses of architectural descriptions

- Expression of system and its evolution
- Communication among stakeholders
- Evaluation and comparison of architectures in consistent manner
- Planning, management and executing activities of system development
- Expression of persistent characteristics of system and supporting principles to guide acceptable evolution
- Verification of systems implementation's compliance with architectural description
- Recording contributions to the body of knowledge of software-intensive system architecture

[IEEE standard 1471-2000]



7 Rules for sound SA documentation

1. Write from the viewpoint of the reader
2. Avoid unnecessary repetition
3. Avoid ambiguity
4. Use a standard organisation
5. Record rationale
6. Keep document current (but not too current)
7. Review document for fitness of purpose



1. Write from the readers point of view

- Documents written for the reader will be read: documents written for the convenience of the writer will not.
- Corollaries
 - Organise documentation for ease of reference
 - Mark TBD (To Be Determined) what you don't know rather than leaving it blank



2. Avoid unnecessary repetition

- Information should be recorded in a single place
 - Makes the change much easier
- The reader should not need to wonder:
 - Is this difference intentional?
 - What is the meaning of the difference?

- Synonymous terms are easily problematic



3. Avoid ambiguity

- If the reader misunderstands – documentation has failed
- Explain your notation!
 - Are the boxes supposed to be modules, objects, classes, processes, functions, procedures, processors,...?
 - Do the arrows means submodule, inheritance, synchronization, exclusion, calls, uses, data flow,...?
- Include a key to the symbols used
 - Box-and-line diagrams are certainly not architectures
 - Don't claim it's anything more than a start at an architecture
 - Ask what boxes and lines *precisely* connote
- Concepts: does the "system" always refer to the same



4. Use standard organisation

- Easier for the reader to find information
- Easier for the writer to know where to put information
- Helps to see if something major is missing



5. Record rationale

- When documenting a result of a decision
 - record the alternatives you rejected
 - and state why
- You won't remember all your reasoning after a few weeks if its not clearly documented!
- How would anyone else have a clue?



6. Keep the document current, but not too current

- Document that is incomplete or out of date is not used
- Revising document to reflect decisions that will not persist is unnecessary expense
 - Decisions that change daily
 - Models and other material mainly useful in the design process, not likely to be updated later
 - If you want traceability, changes must be propagated throughout the document, e.g., from requirements to architecture and further to implementation



7. Review document for the fitness of purpose

- Do the stakeholders get answers to their concerns?
- What is the purpose of a particular model or drawing? What questions does it answer?
- Have the document reviewed by representatives of intended readers
 - Does it contain the right information?
 - Does it present the information in a useful way?
 - Does it satisfy their needs?



SA documentation package

*Documenting a software architecture
is a matter of documenting the relevant views, and
then adding information
that applies to more than one view.*

- View packets
- Documentation beyond views



SA documentation package

from ECS (Earth Observing Science Data...System) example in Clements et al. DSA, 2002

- Two volumes
 - Volume I contains information that applies to more than one view
 - Volume II contains the architectural views for the system (view packets)



Volume I contents

- Architecture documentation roadmap
- System overview
- Software architecture view template
- Mapping between views
- Directory
- Architecture glossary and acronym list
- Rationale, background and design constraints



SA view template: standard organisation

1. Primary representation
2. Element catalogue
3. Context diagram
4. Variability guide
5. Architecture background
6. Other information
7. Related view packets



1. Primary representation

- Shows the elements and their relationships that populate the view packet
- Usually graphical
 - with the meaning of the notation explained !!!



2. Element catalogue

- Elements and their properties
 - Names each element in the view packet and lists its properties
 - Relations and their properties
 - Relations and exceptions not shown in the primary view are recorded here
 - Element interface
 - Element behaviour
- } if relevant



5. Architecture background

- Rationale
 - Design decisions reflected in the view packet
 - Rejected alternative and reasons for rejection
- Analysis results
 - E.g., performance analyses conducted
- Assumptions
 - E.g., about environment and need
 - Invariants of the environment, tools, skills available
 - Why the design provided is sufficient for what's needed



Volume II: SA views, examples

1. Module Decomposition View (MDV)
 - 1.1. MDV packet 1: The ECS System
 - 1.2. MDV packet 2: The Science Data Processing Segment
 - 1.3. MDV packet 3: The Client Subsystem
 - ...
2. Module Uses View
3. Module Generalization View
- ...
10. Allocation Work Assignment View



Conclusions

- Documentation is needed to communicate architecture
- Think about how you document
 - 7 rules of sound documentation
 1. Write from the viewpoint of the reader
 2. Avoid unnecessary repetition
 3. Avoid ambiguity
 4. Use a standard organisation
 5. Record rationale
 6. Keep document current (but not too current)
 7. Review document for fitness of purpose
- Document views and what applies to more than one view
- Think about the purpose of the document (or part of it) – if you don't know it, you may as well not document

