

IV: Architectural Design: 9, 10: Criteria, Components.

Results

- ▶ Conference administration (Chapter 19)
- ▶ Usable: different volunteers; many over time
- ▶ Portable: used at each conference in different venues
- ▶ Correct: must fulfill the specification as the local users cannot identify and solve errors
- ▶ Comprehensible: must facilitate easy changes of the system
- ▶ Efficient: simple system for administrative tasks
- ▶ Interoperable: stand-alone system

<i>Criterion</i>	<i>Very important</i>	<i>Important</i>	<i>Less important</i>	<i>Irrelevant</i>	<i>Easily fulfilled</i>
Usable	X				
Secure			X		
Efficient					X
Correct		X			
Reliable			X		
Maintainable			X		
Testable			X		
Flexible			X		
Comprehensible		X			
Reusable			X		
Portable	X				
Interoperable				X	

- ▶ The rest: less important because of the nature of the problem and application domains + the stable nature of these domains

Architectural design has two views:

Component Architecture

Static .

Classes, stable aspects, related components, logical level and structure for descriptions.

Process Architecture

Dynamic: what is going on over time.

Objects, dynamic aspects, coordination of processes, physical level and structure for execution.

Principles

- Define and prioritize criteria. (Focus on usability or something else)
- Bridge criteria and technical platform.
- Evaluate design early.

Objects in Analysis and Design

► Analysis:

- Phenomena outside the computer system
- Identity: identifies an object
- State: the qualities that characterise an object
- Behavior: the events an object have performed or suffered

► Design (and programming):

- Phenomena inside the computer system
- Identity: gets access to an object
- State: the values of the object's attributes and object structures
- Behavior: the operations an object can perform on request and offers to other objects (methods)

Criteria:

- General criteria. (**Usable**, secure, efficient, correct, reliable, maintainable, testable, **flexible**, **comprehensible**, reusable, portable and interoperable) (Which us these will we focus on?) (Correct = consistent based on the requirements. Fulfils the requirements)
- Specific criteria in OOA&D:
 - Usability
 - The systems as a whole
 - The users' needs
 - The technical platform
 - Flexibility
 - Consequences of changes
 - Modular design
 - Comprehensibility
 - Overview
 - Abstraction
 - Use of patterns

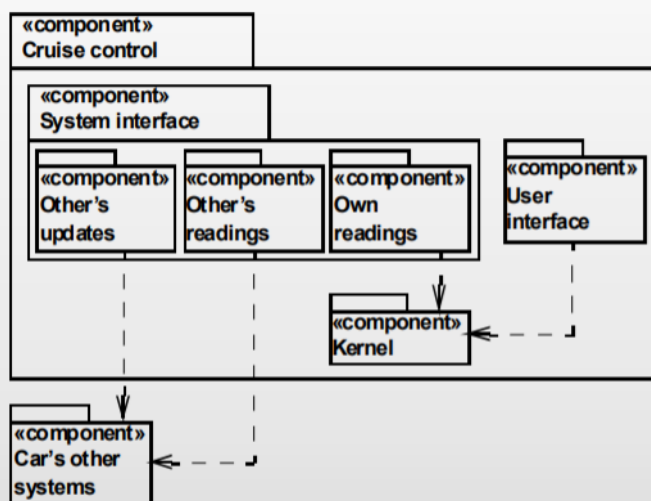
Typical conditions for design of a system's architecture

Technical	<ul style="list-style-type: none">• Existing hardware, basic software, and systems.• Reuse of patterns and existing components.• Use of purchased standard components.
Organizational	<ul style="list-style-type: none">• Contractual arrangements.• Plans for continued development.• Division of work between developers.
Human	<ul style="list-style-type: none">• Design competences.• Experience with similar systems.• Experience with technical platform.

Criteria: Summary

Purpose	<ul style="list-style-type: none">To set design priorities.
Concepts	<ul style="list-style-type: none">Criterion: A preferred property of an architecture.Conditions: The technical, organizational, and human opportunities and limits involved in performing a task.
Principles	<ul style="list-style-type: none">A good design has no major weaknesses.A good design balances several criteria.A good design is usable, flexible, and comprehensible.
Results	<ul style="list-style-type: none">A collection of prioritized criteria.

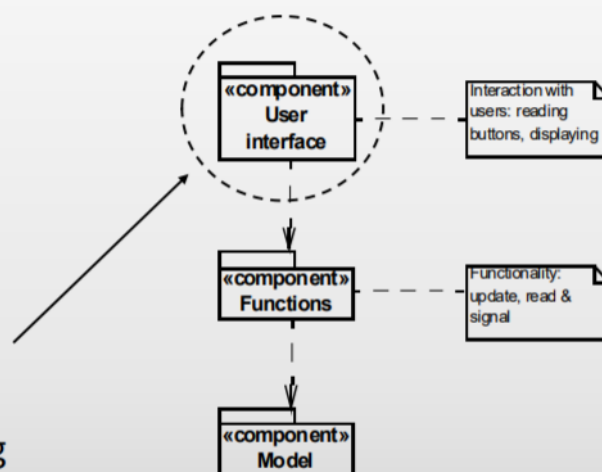
Components activity



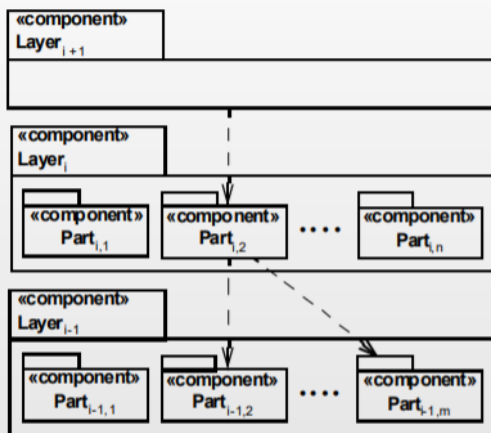
- ▶ A structural perspective
- ▶ Separates concerns in a system
- ▶ Emphasizes comprehensibility and flexibility

Key Concept: Component

- ▶ A collection of program parts
- ▶ Constitutes a totality
- ▶ Has a well-defined responsibility
- ▶ Smallest: a class
- ▶ Largest: a system
- ▶ Example:
This component has the responsibility for reading the buttons and updating the display



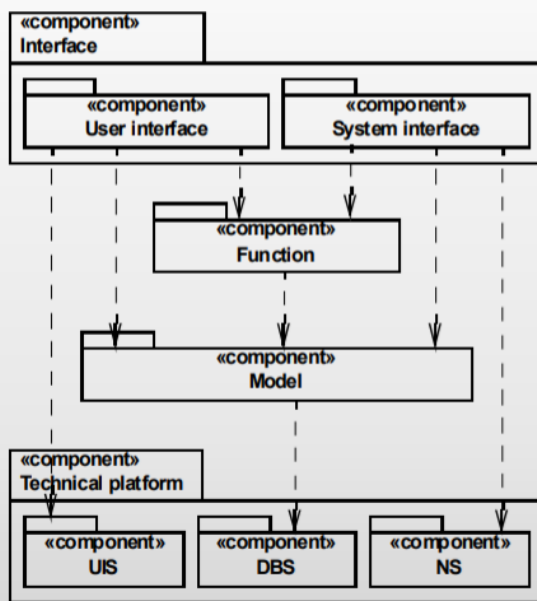
Pattern: The Layered Architecture



- ▶ Layer: describes a component's responsibility by the operations it provides to a layer above and those that are applied from the layer below
- ▶ Part: no substantial interaction with other parts in the same layer
- ▶ Closed architecture: only apply operations from an adjacent layer
Open architecture: apply operations from any other layer
- ▶ Strict architecture: only apply operations from a layer below
Relaxed architecture: apply operation from layer both above and below

You can mix the two: Open architecture + relaxed = spaghetti code.

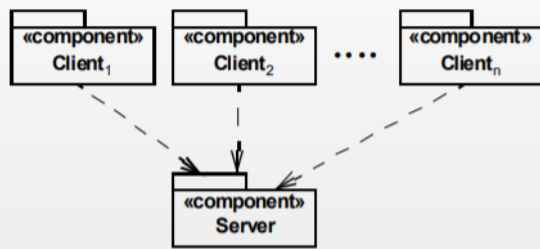
Pattern: The Generic Architecture



- ▶ The generic architecture reflects the division of the context into problem domain and application domain
- ▶ “Technical platform” is an extension and encapsulation of the underlying technical platform
- ▶ Example: a single Dankort terminal

NS = network system, DBS = Database system, UIS = user interface system.

Pattern: Client-Server Architecture



Client	Server	Architecture
U	U + F + M	Distributed presentation
U	F + M	Local presentation
U + F	F + M	Distributed functionality
U + F	M	Centralised data
U + F + M	M	Distributed data

- ▶ Originally for distribution of physically (geographically) dispersed processors
- ▶ Can also be used logically, independently of processors
- ▶ One server and a number of clients
- ▶ Clients are assigned to the server dynamically
- ▶ The distribution can be based on various divisions between server and clients
- ▶ Example: the Dankort operator (Nets) and the shops

U = user interface, F = functions, M = model.

It is possible to have a distributed data and for the server to also have functions. But then one would have a subset of functions.