DESIGN: ARCHITECTURE, CRITERIA AND COMPONENTS

SU:E15:L9

October 19, 2015

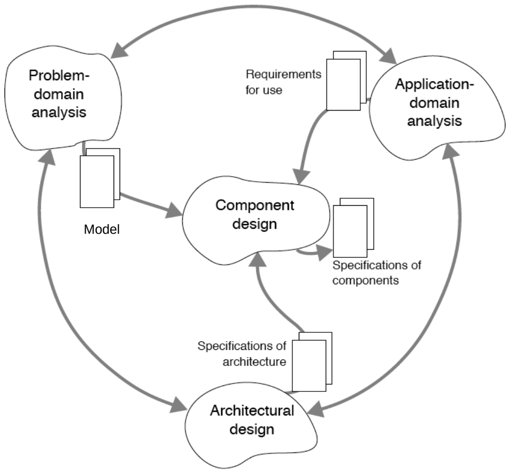
CONTENTS

- **Design: Criteria**
- **Design: Component architecture**

1

ARCHITECTURAL DESIGN

ACTIVITIES IN 'ARCHITECTURAL DESIGN'



Problem-domain analysis
Requirements for use
Application-domain analysis
Model
Component design
Specifications of components
Specifications of architecture
Architectural design

Criteria

- What are the conditions and criteria for design?
- Criteria

Component architecture

- How is the system structured into components?
- Component architecture and component

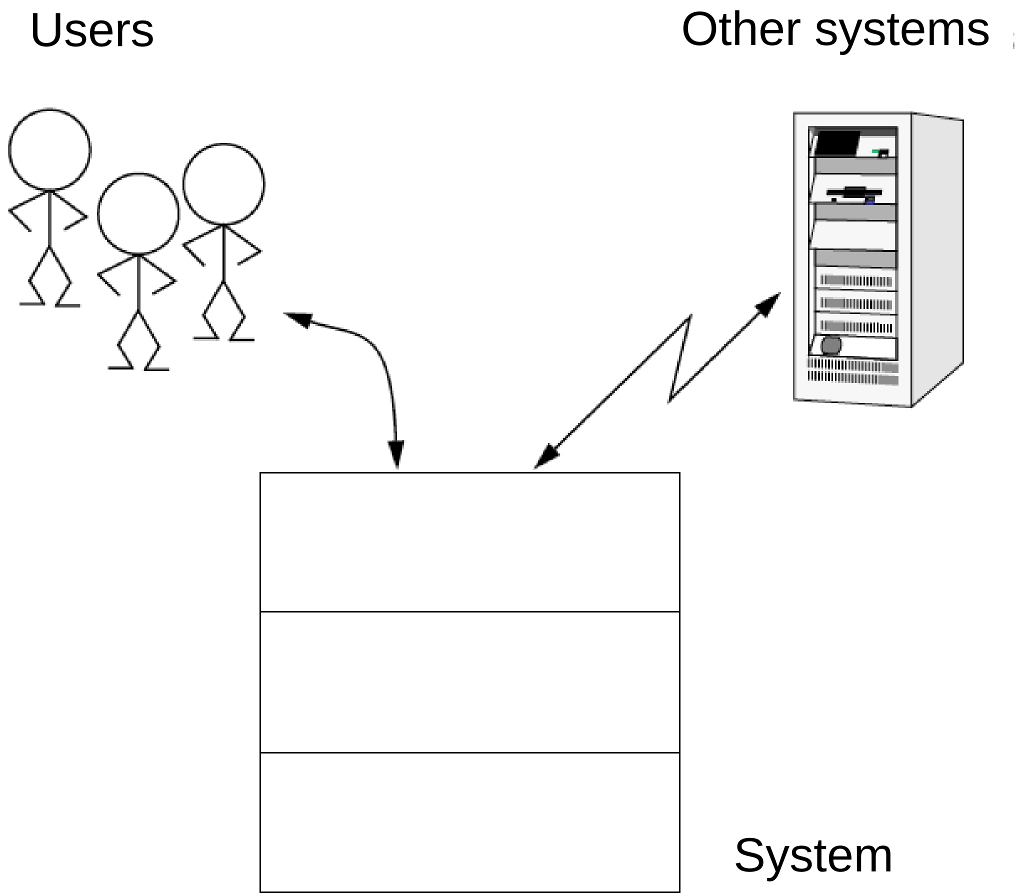
Process architecture

- How are the system's processes distributed and coordinated?
- Process architecture and process

THE CONTEXT AND THE ARCHITECTURE

Architecture: A general basic structure, which is extended later.

IT-system: A collection of components that implement modeling requirements, functions and interfaces.



ARCHITECTURAL DESIGN

- **Criterion:** a preferred property of an architecture.

  What are the conditions for criteria and design?

- **Component architecture:** a system structure composed of interconnected components.

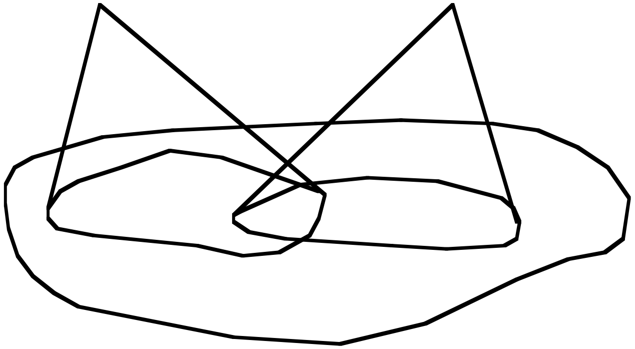
  How is the system structured into components?

- **Process architecture:** a system-execution structure composed of interdependent processes.

  How are the system's processes distributed and coordinated?

Component Architecture
- Classes
- Stable aspects
- Related components
- Logical level
- Structure for descriptions

Process Architecture
- Objects
- Dynamic aspects
- Coordination of processes
- Physical level
- Structure for execution



The system

Principles:
- Define and prioritize criteria
- Bridge criteria and technical platform
- Evaluate designs early

SU:E15:L9

ANALYSIS AND DESIGN

	Analysis	Design
Object	Phenomenon in the context of the IT-system	A part of the system; Some of the objects represent parts of the reality
Class	Behavior is described in abstract patterns of events.	Behavior is described in a collection of operations.

OVERVIEW OF ‘ARCHITECTURAL DESIGN’

Purpose

- To structure a computerized system.

Concepts

- **Criterion**: A preferred property of an architecture.
- **Component architecture**: A system structure composed of interconnected components.
- **Process architecture**: A system-execution structure composed of interdependent processes.

Principles

- Define and prioritize criteria.
- Bridge criteria and technical platform.
- Evaluate designs early.

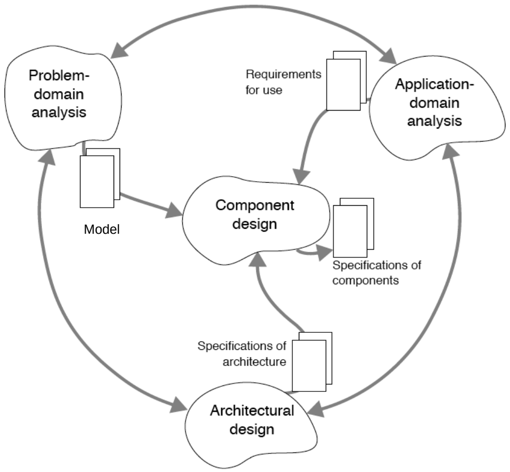
Results

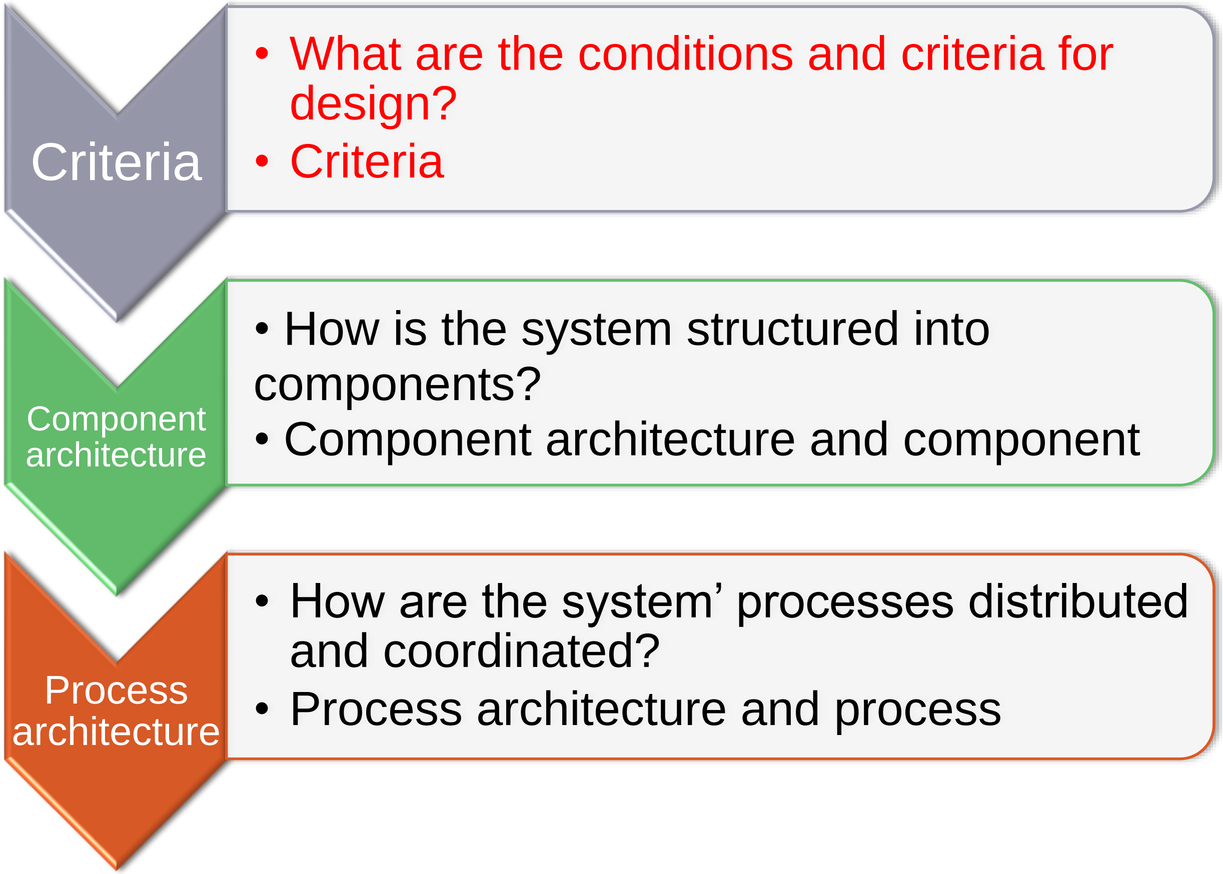
- Structures for a system’s components and processes

2

CRITERIA

ACTIVITIES IN 'ARCHITECTURAL DESIGN'


Problem-domain analysis
Requirements for use
Application-domain analysis
Component design
Model
Specifications of components
Specifications of architecture
Architectural design



Criteria

- What are the conditions and criteria for design?
- Criteria

Component architecture

- How is the system structured into components?
- Component architecture and component

Process architecture

- How are the system' processes distributed and coordinated?
- Process architecture and process

Criterion: a preferred property of an architecture

What are the conditions for criteria and design?

OVERVIEW OF ‘CRITERIA’

Purpose

- To set design priorities.

Concepts

- **Criterion**: A preferred property of an architecture.
- **Conditions**: the technical, organizational, and human opportunities and limits involved in performing a task.

Principles

- A good design has no major weaknesses.
- A good design balances several criteria.
- A good design is usable, flexible, and comprehensive.

Results

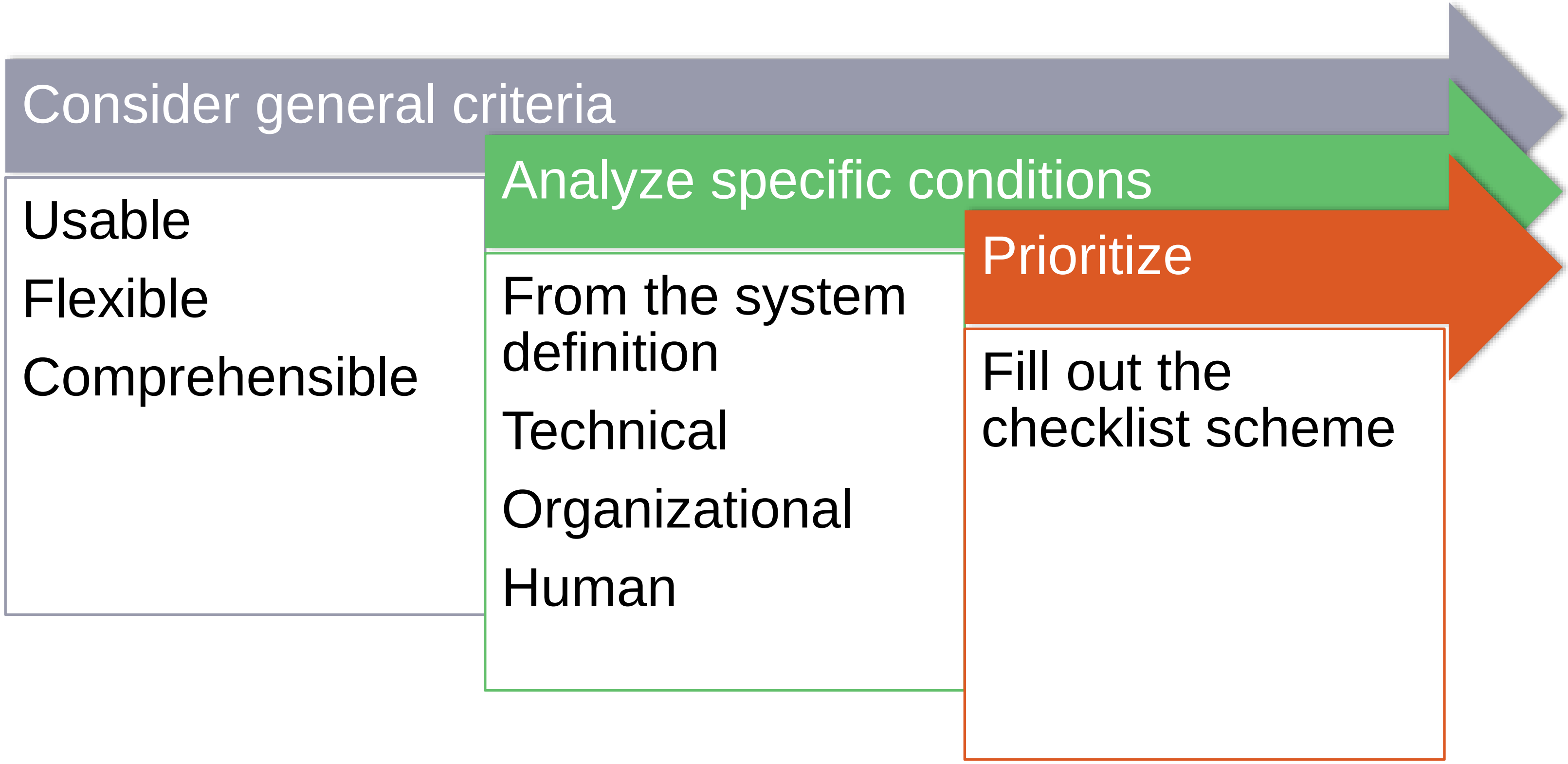
- A collection of prioritized criteria.

RESULT OF ’CRITERIA’

- **Prioritization of criteria for a design of a system:**

	Very important	Important	Less important	Irrelevant	Easily fulfilled
Usable	X				
Secure			X		
Efficient					X
Correct		X			
Reliable			X		
Maintainable			X		
Testable			X		
Flexible			X		
Comprehensive		X			
Reuseable			X		
Portable	X				
Interoperable				X	

ACTIVITIES IN ‘CRITERIA’



Consider general criteria

Usable

Flexible

Comprehensible

Analyze specific conditions

From the system definition

Technical

Organizational

Human

Prioritize

Fill out the checklist scheme

CONSIDER CRITERIA

CLASSICAL CRITERIA

Useable: the system's adaptability to the organizational, work-related, and technical context

Secure: the precautions against unauthorized access to data and facilities

Efficient: the economical exploitation of the technical platform's facilities

Correct: the fulfillment of requirements

Reliable: the fulfillment of the required precision in function execution

Maintainable: the cost of locating and fixing system defects

Testable: the cost of ensuring that the deployed system performs its intended function

Flexible: the cost of modifying the deployed system

Comprehensible: the effort needed to obtain a coherent understanding of the system

Reusable: the potential for using system parts in other related systems

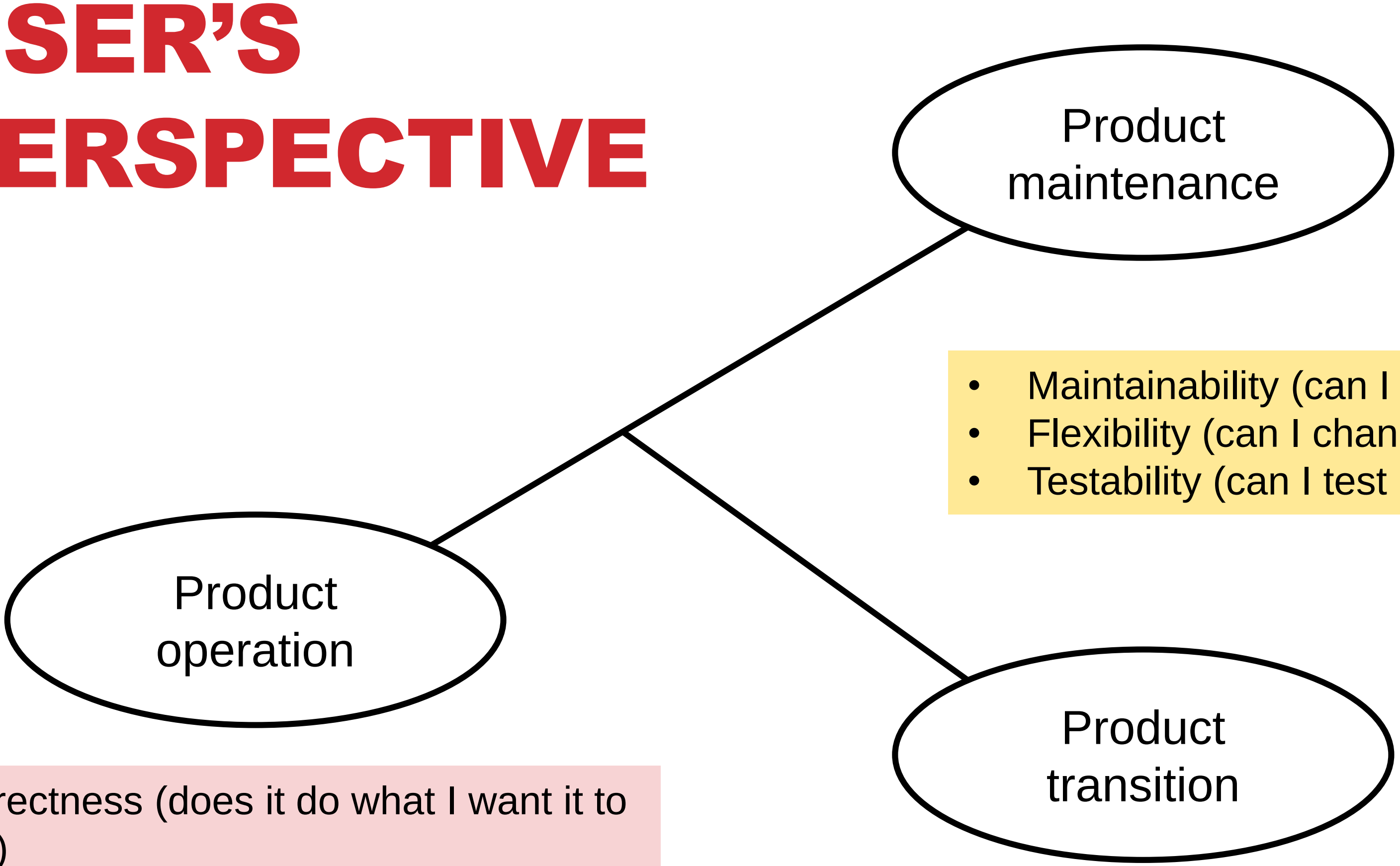
Portable: the cost of moving the system to another technical platform

Interoperable: the cost of coupling the system to other systems

GENERAL CRITERIA

- **Usable**
  How it works in the context
  - Users' needs
  - The technical platform
  - ➢ Base the design on experiments
- **Flexible**
  Our knowledge is incomplete
  - Consequences of future changes in the context?
  - ➢ Modular design (encapsulations)
- **Comprehensible**
  Must be easy to understand
  - ➢ Explore and combine many technical possibilities
  - ➢ Clarity and coherence
  - ➢ Abstraction
  - ➢ Reuse of patterns
  - ➢ Group responsibilities

USER'S PERSPECTIVE



- Maintainability (can I fix it?)
- Flexibility (can I change it?)
- Testability (can I test it?)

- Correctness (does it do what I want it to do?)
- Reliability (does it do it accurately all the time?)
- Efficiency (will it run on my hardware as well as it can?)
- Integrity (is it secure?)
- Usability (can I operate it?)

- Portability (will I be able to use it on another machine?)
- Reusability (will I be able to reuse some of the software?)
- Interoperability (will I be able to interface it will some other system?)

ANALYZE SPECIFIC CONDITIONS

Typical conditions for design of an architecture

Technical	• Existing hardware, basic software, and systems. • Reuse of patterns and existing components. • Use of purchased standard components.
Organizational	• Contractual arrangements. • Plans for continued development. • Division of work between developers.
Human	• Design competence. • Experience with similar systems. • Experience with technical platform.

The conditions must be identified, discussed and evaluated.

creates an important focus in design

PRIORITIZE

- **Make a well considered and clear prioritization of the general criteria**

	Very important	Important	Less important	Irrelevant	Easily fulfilled
Usable					
Secure					
Efficient					
Correct					
Reliable					
Maintainable					
Testable					
Flexible					
Comprehensive					
Reuseable					
Portable					
Interoperable					

- **Add specific conditions**

OVERVIEW OF ‘CRITERIA’

Purpose

- To set design priorities.

Concepts

- **Criterion**: A preferred property of an architecture.
- **Conditions**: the technical, organizational, and human opportunities and limits involved in performing a task.

Principles

- A good design has no major weaknesses.
- A good design balances several criteria.
- A good design is usable, flexible, and comprehensive.

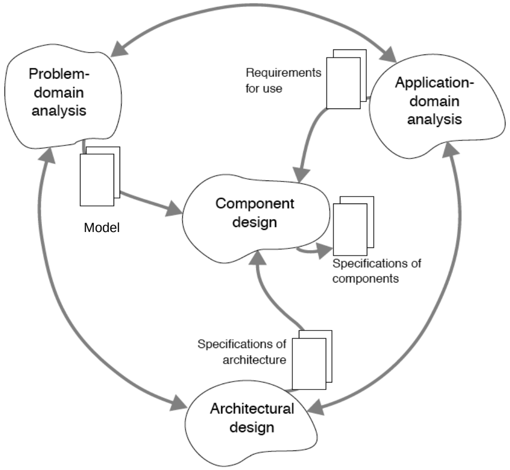
Results

- A collection of prioritized criteria.

1

COMPONENTS

ACTIVITIES IN 'ARCHITECTURAL DESIGN'


Problem-domain analysis
Requirements for use
Application-domain analysis
Model
Component design
Specifications of components
Specifications of architecture
Architectural design

Criteria

- What are the conditions and criteria for design?
- Criterion

Component architecture

- How is the system structured into components?
- Component architecture and component

Process architecture

- How are the system' processes distributed and coordinated?
- Process architecture and process

OVERVIEW OF ‘COMPONENTS’

Purpose

- To create a comprehensive and flexible system structure.

Concepts

- **Component architecture**: A system structure of interconnected components.
- **Component:** A collection of program parts that constitutes a whole and has well-defined responsibilities.

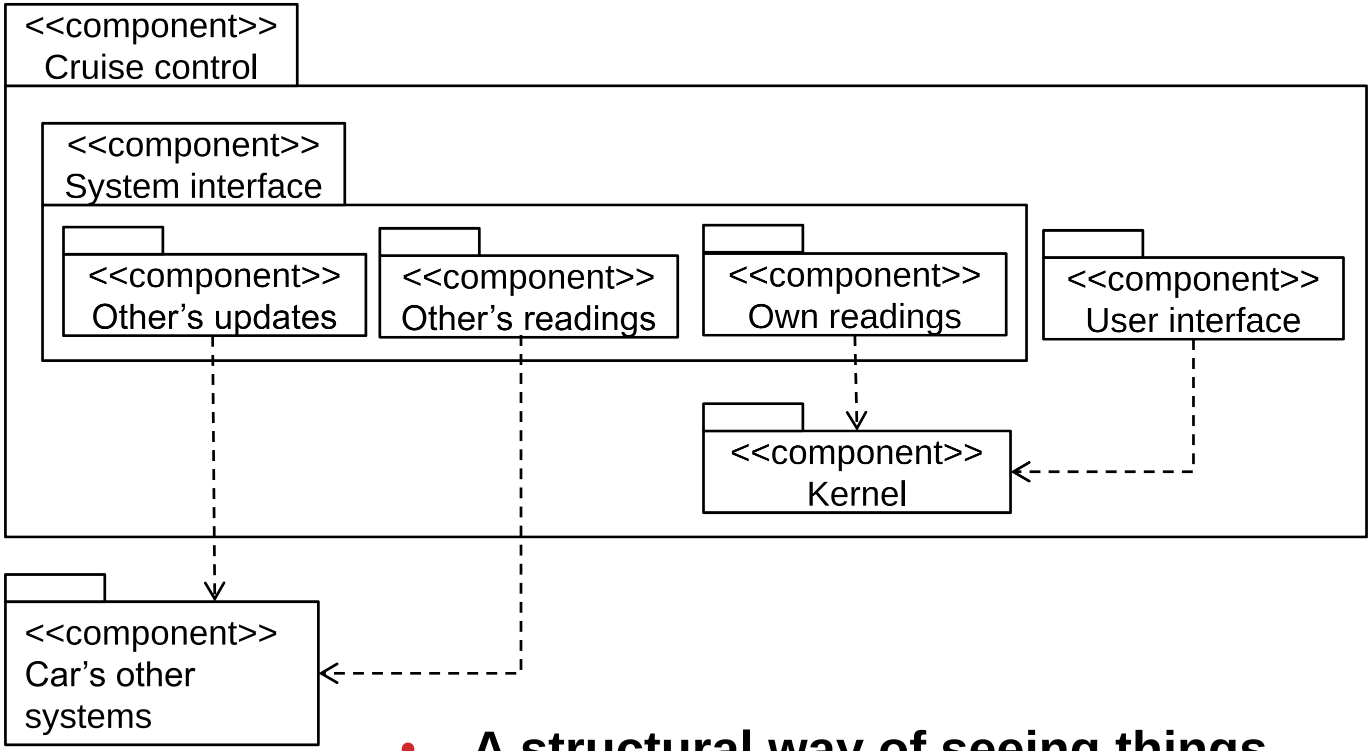
Principles

- Reduce complexity by separating concerns.
- Reflect stable context structures.
- Reuse existing components.

Results

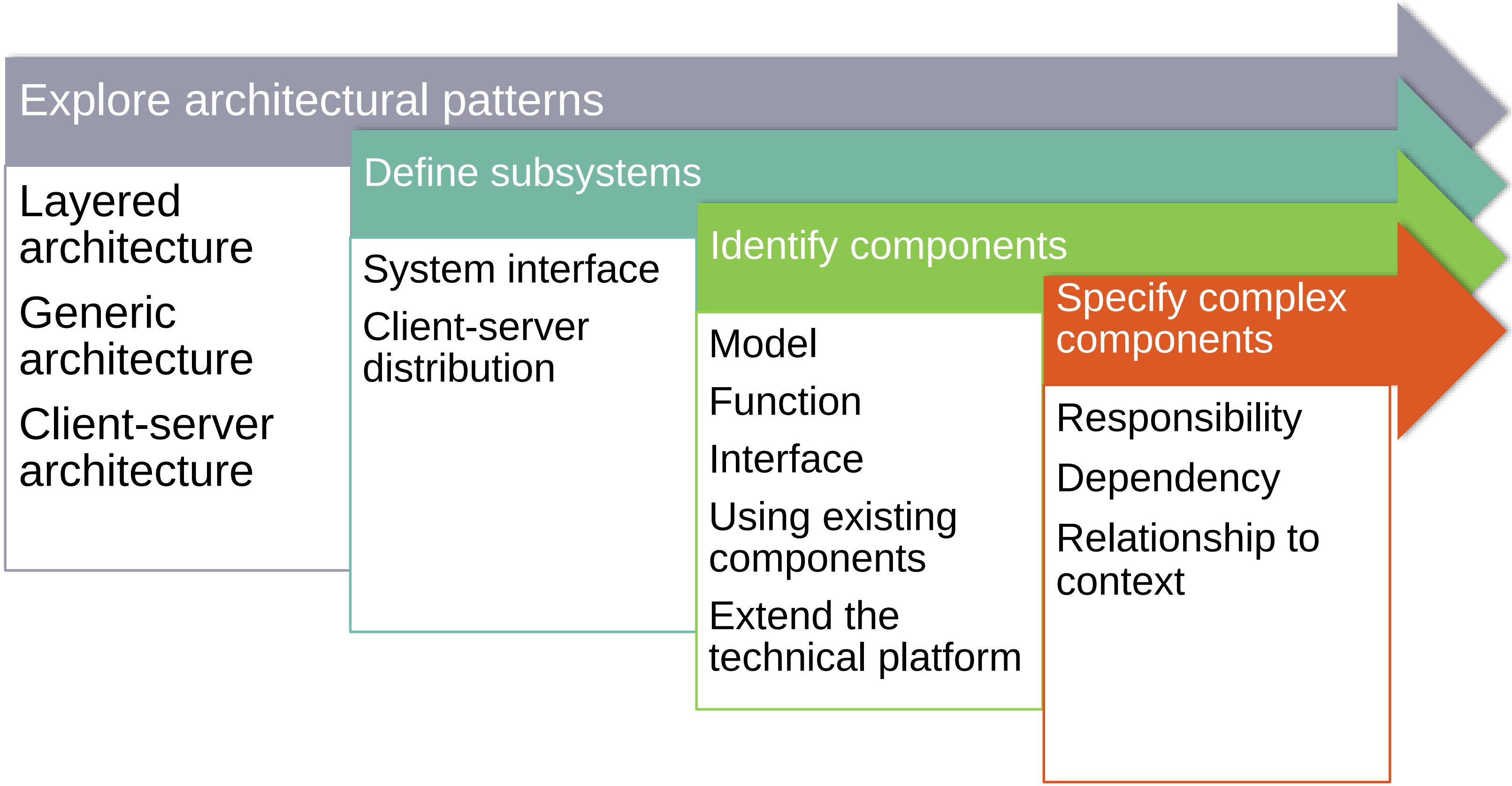
- A class diagram with specifications of the complex components.

RESULT OF 'COMPONENTS'



- **A structural way of seeing things**
- **Divides the concerns of a system**
- **Directs attention to comprehensiveness and flexibility**

ACTIVITIES IN ‘COMPONENTS’



Explore architectural patterns

- Layered architecture
- Generic architecture
- Client-server architecture

Define subsystems

- System interface
- Client-server distribution

Identify components

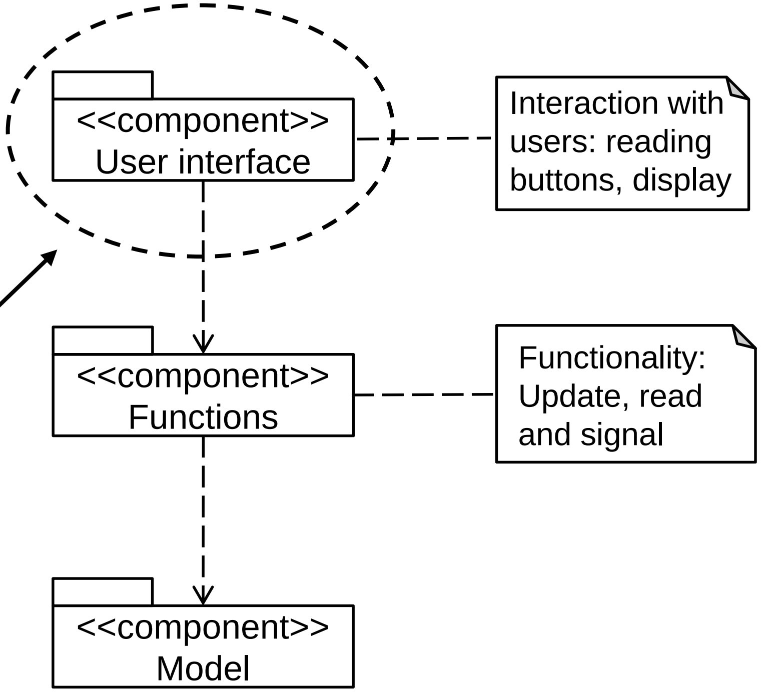
- Model
- Function
- Interface
- Using existing components
- Extend the technical platform

Specify complex components

- Responsibility
- Dependency
- Relationship to context

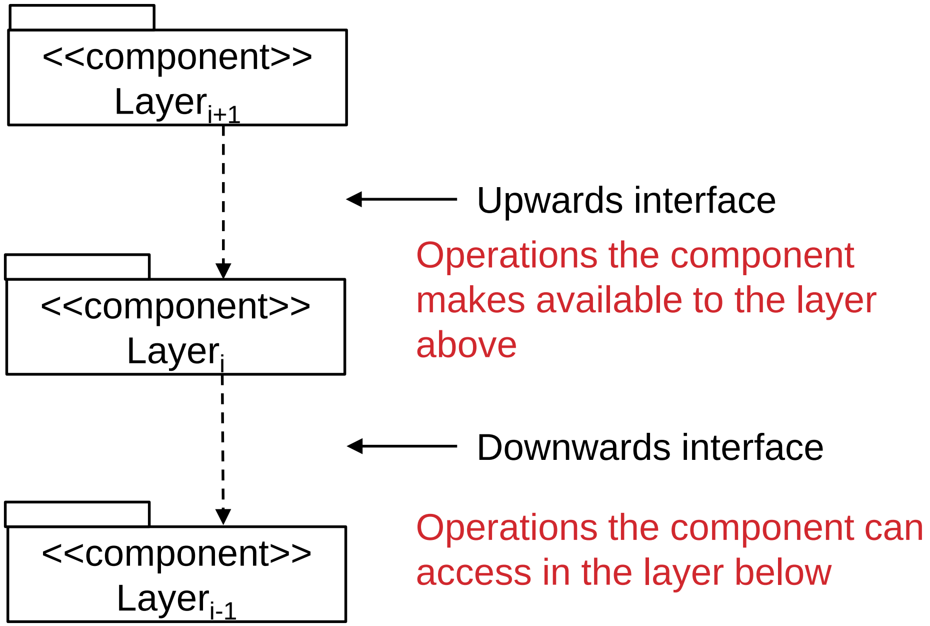
COMPONENT

- **A collection of program parts**
- **Constitutes a whole**
- **Has clear and well-defined responsibilities**
  - Smallest: one class
  - Largest: a system
- **Example:**
  The component is responsible for reading buttons and updating the display



<<component>>
User interface
Interaction with users: reading buttons, display
<<component>>
Functions
Functionality: Update, read and signal
<<component>>
Model

PATTERN: LAYERED ARCHITECTURE



Operations the component makes available to the layer above

Operations the component can access in the layer below

- **Layer:**

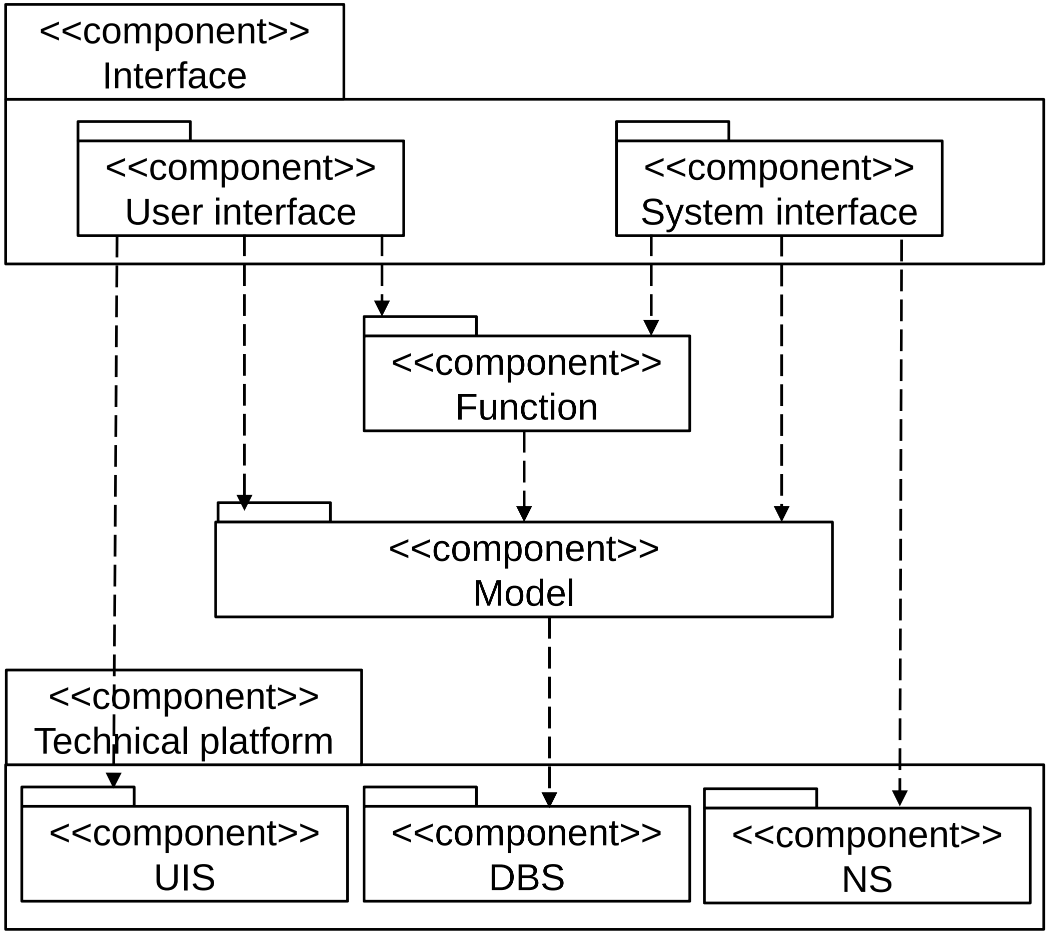
  Describes a component’s responsibility by showing which operations are offered from above and which are used from below

- **Dependency**:

  Dashed arrows denote dependencies

  A dependency implies that a change in one component (pointed at) may affect the other component (pointed from)

PATTERN: GENERIC ARCHITECTURE



- The generic architecture reflects the division between the problem domain and application domain
  - **Model component**: implements the problem-domain model
  - **Function layer**: contains system functions
  - **Interface**: can be decomposed into two separate parts: UI and SI
- “The technical platform” is an extension and encapsulation of the underlying technical platform

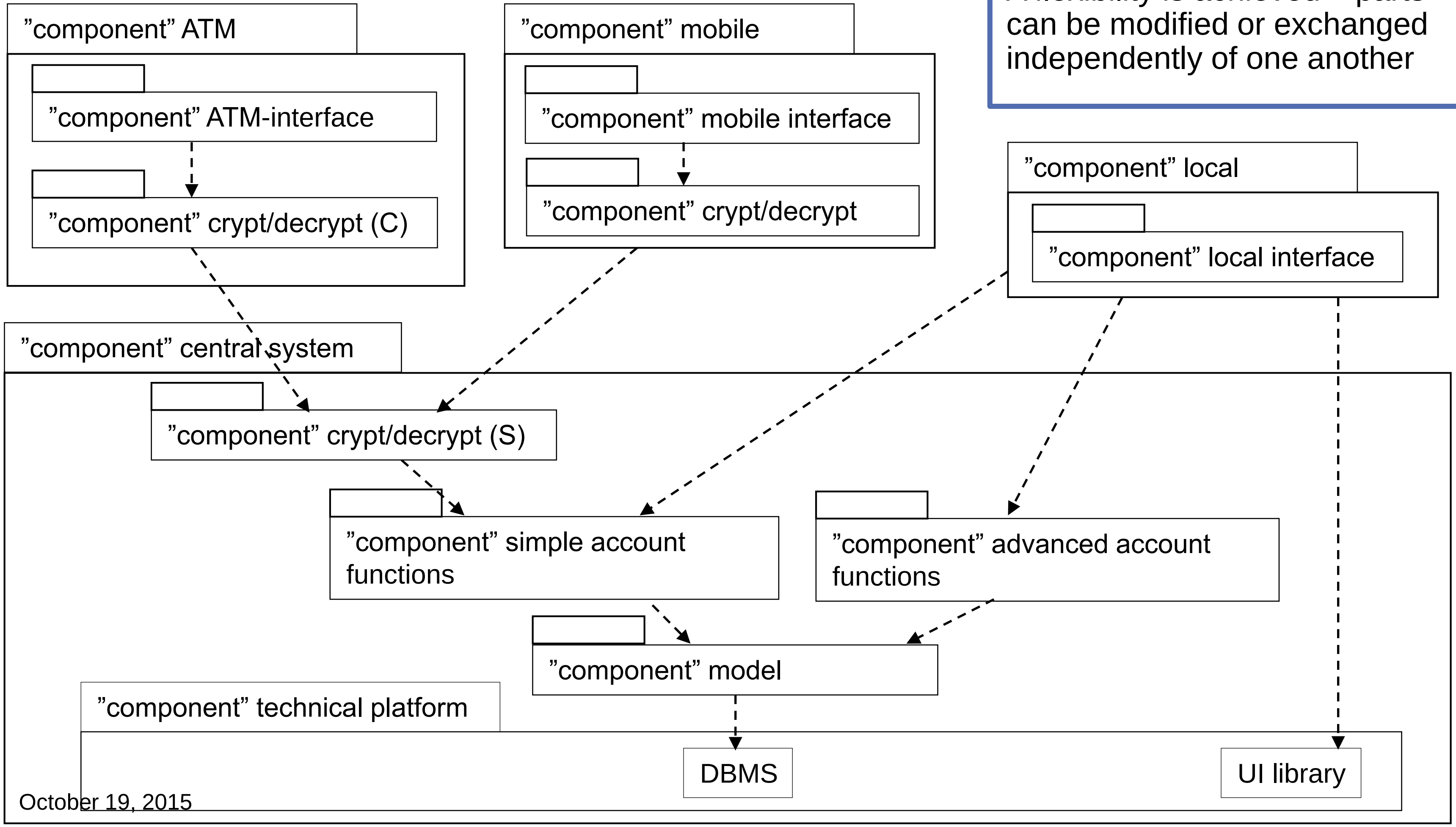
PATTERN: LAYERS AND SUBCOMPONENTS (PARTS)

Part:

Adds a horizontal decomposition to the vertical decomposition with

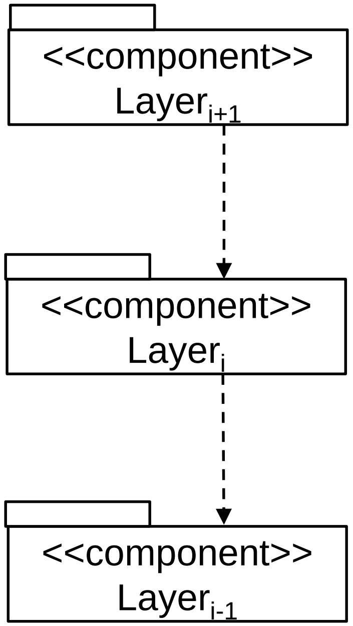
There should be no significant interaction with the other parts in the same layer

A flexibility is achieved – parts can be modified or exchanged independently of one another



”component” ATM
”component” ATM-interface
”component” crypt/decrypt (C)
”component” mobile
”component” mobile interface
”component” crypt/decrypt
”component” local
”component” local interface
”component” central system
”component” crypt/decrypt (S)
”component” simple account functions
”component” advanced account functions
”component” model
”component” technical platform
DBMS
UI library

VARIATIONS

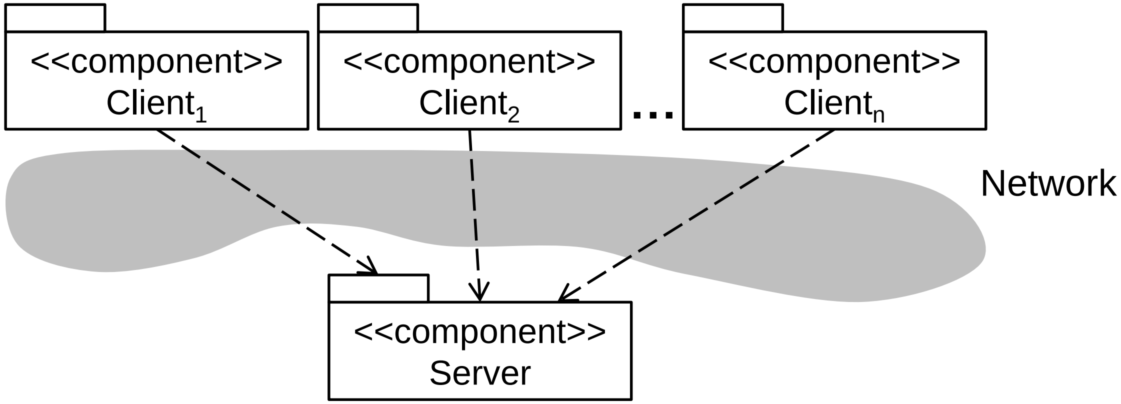


- **Closed architecture:** can only use operations from the layer immediately above or below
- **Open architecture:** can use operations from any layer above or below
- **Strict architecture:** can only use operations in layers below
- **Relaxed architecture:** can use operations in layers below and above

Closed-strict	Open-strict
Closed-relaxed	Open-relaxed

Many layers a closed-strict may be best
Few layers an open-strict may be best

PATTERN: CLIENT-SERVER ARCHITECTURE



Originally developed to handle the distribution of a system among several geographically dispersed processors

Overlaps and supplements the generic architecture

Variations regarding distribution of tasks between client and server

- A generic architecture to get several computers to communicate with a central computer
  - The components: a server and several clients
  - The server has a collection of operations that it makes available to the clients
- Asymmetric

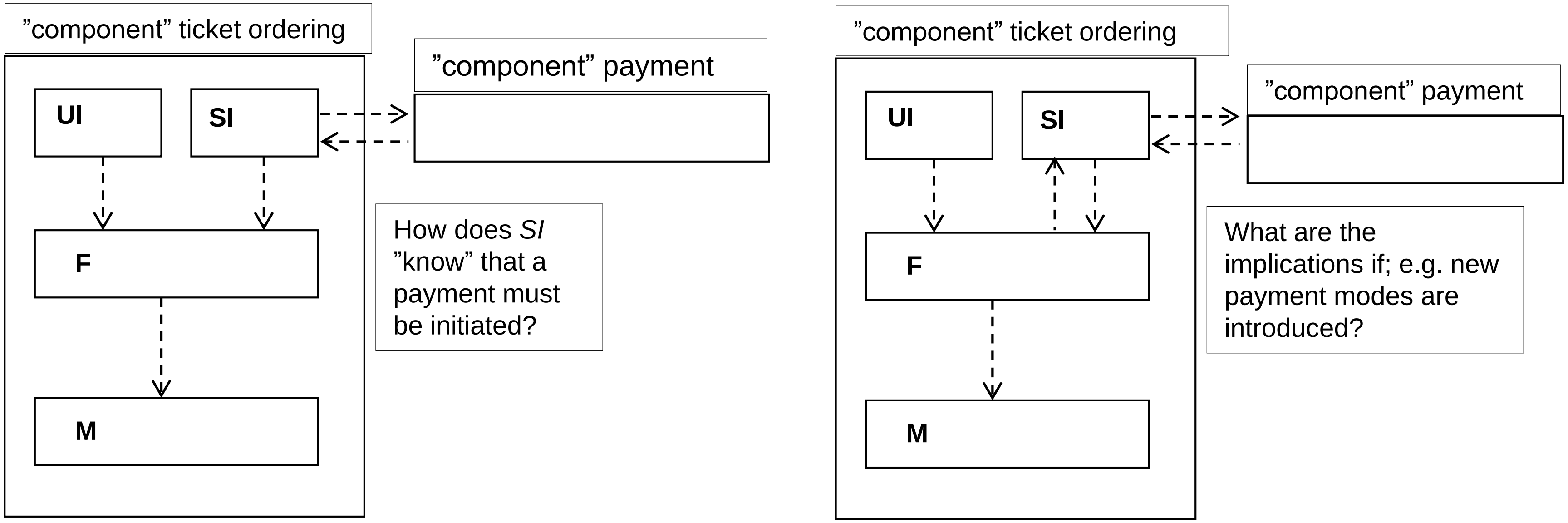
VARIATIONS OF THE CLIENT SERVER ARCHITECTURE

- Based on the client server and the general patterns
- Five component architectures with varying degrees of distribution

Klient	Server	Arkitektur
I	I + F+ M	Distributed presentation
I	F + M	Local presentation
I + F	F + M	Distributed functionality
I + F	M	Centralized data
I + F + M	M	Distributed data

DEFINE SUBSYSTEMS

- Large systems must be decomposed (several independent subsystems)
- Each subsystem has its own architecture
  - Based on the generic architecture pattern
- A subsystem’s “System interface” component provides other sub systems with a coherent interface for accessing the given subsystem’s functionality



How does *SI* ”know” that a payment must be initiated?

What are the implications if; e.g. new payment modes are introduced?

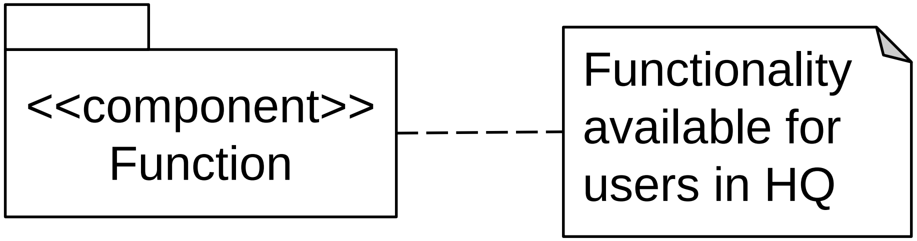
Alternatives

IDENTIFY COMPONENTS

Concern	Model	Function	Interface
Responsibility	The problem domain model	The functionality on the model	The interaction between functionality and users or other systems
Contextual issues	In cohesive or complex problem domains	Need for in cohesive or complex functionality	In cohesive or complex usage; in cohesive or complex actors
Exemplars	Accounting, reservations, inventory, administration	Payroll, signal processing, cruise control, prediction monitoring	Browsing, games, presentation monitoring
Special needs	Databases	Model-related functions, application-related functions, cryptography	Screens, windows, buttons, print-outs, devices, communication

- **Use these questions to consider the division into layers and parts**
- **Use existing components**

SPECIFY COMPLEX COMPONENTS - NOTES



Component with note

SPECIFY COMPLEX COMPONENTS – CRC CARDS

- **Describe the component in details:**
  - Responsibility
  - Dependency (for classes with specified collaboration)
  - Relationship to context
- **In a scheme or diagram**

Responsibility	Functionality provided to the uses in the HQ
Dependency	Model Component
Relationship to context	Functions needed by the front office in the HQ

Example: Specification of a function component

OVERVIEW OF ‘COMPONENTS’

Purpose

- To create a comprehensive and flexible system structure.

Concepts

- **Component architecture**: A system structure of interconnected components.
- **Component:** A collection of program parts that constitutes a whole and has well-defined responsibilities.

Principles

- Reducer complexity by separating concerns.
- Reflect stable context structures.
- Reuse existing components.

Results

- A class diagram with specifications of the complex components.