SAA 03 – Drivers and Decisions

Dr. Pablo Oliveira Antonino
pablo.antonino@iese.fraunhofer.de

TU Kaiserslautern, SS2018
Lecture "Software and System Architecture (SSA)"

Discussion

- **RECAP LAST LECTURE**
- Explain the contents of the last lecture
  - What were the topics?
  - Why do we need it?
  - How does it work?
  - How is it created, used, and/or evolved?

© Fraunhofer IESE
@ Jens Knodel

Fraunhofer
IESE

Architecture Drivers

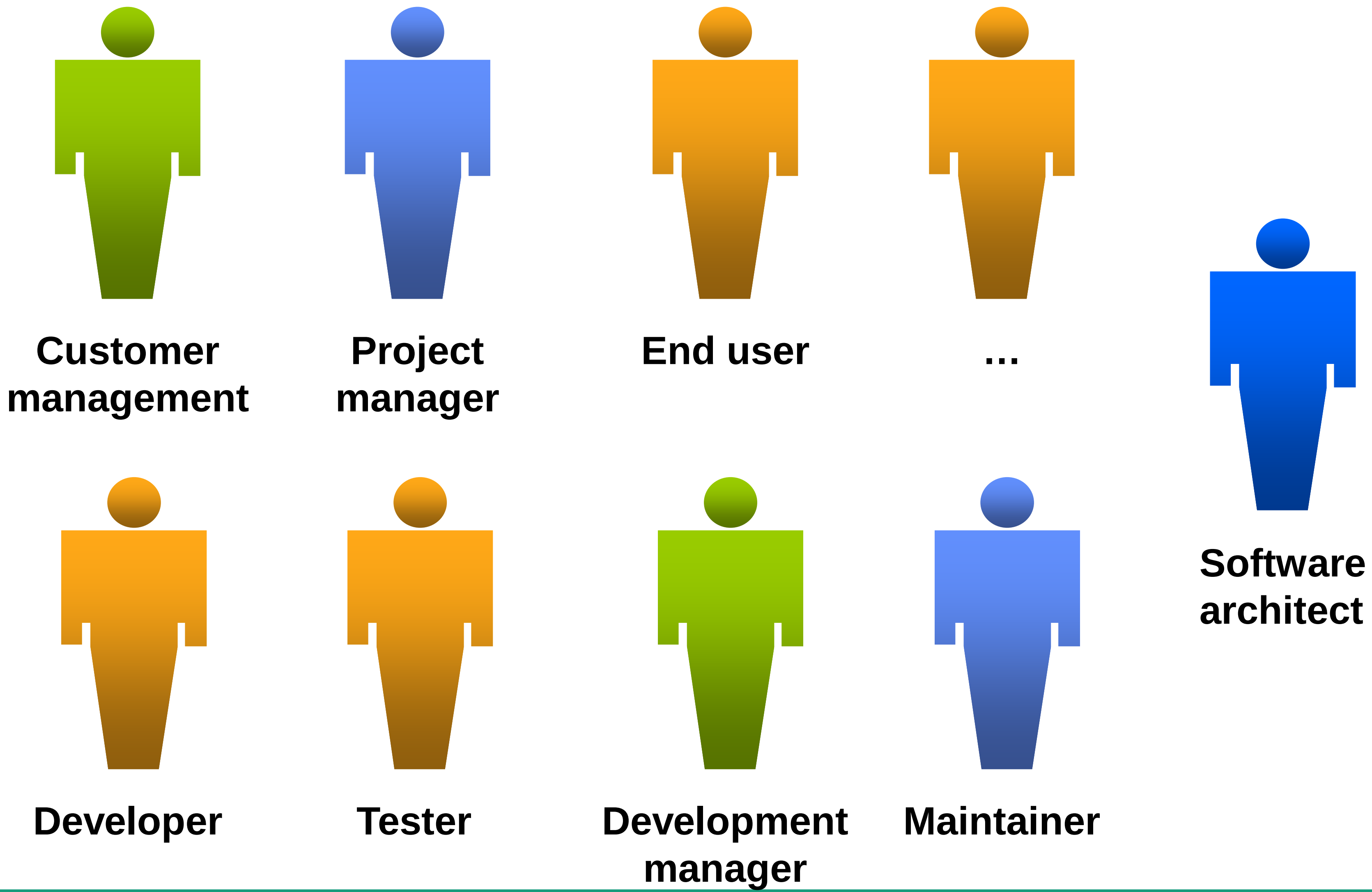
The Role of Stakeholders and their Involvement

- **Stakeholders** have **concerns**
  - Concerns form the product…
  - … and drive the architecture
- The architect has to
  - **Identify** and know the stakeholders!
  - **Involve** the stakeholders early and continuously!
  - Know their **concerns**!
    - Real needs, wishes
  - **Manage their expectations**!
    - Prioritize: not every wish can be fulfilled
    - Make tradeoffs

© Fraunhofer IESE

Fraunhofer IESE

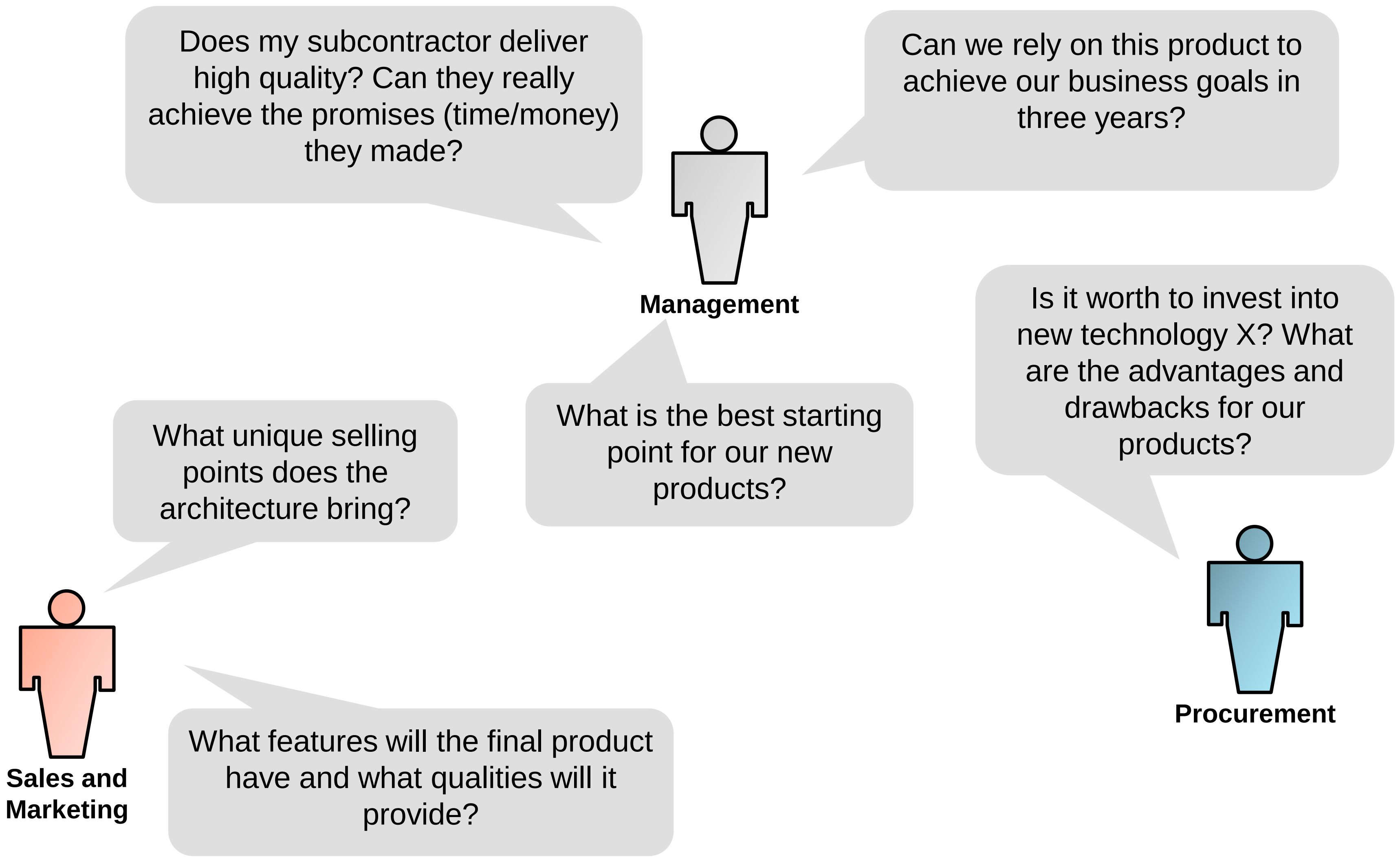
Typical Stakeholders



© Fraunhofer IESE

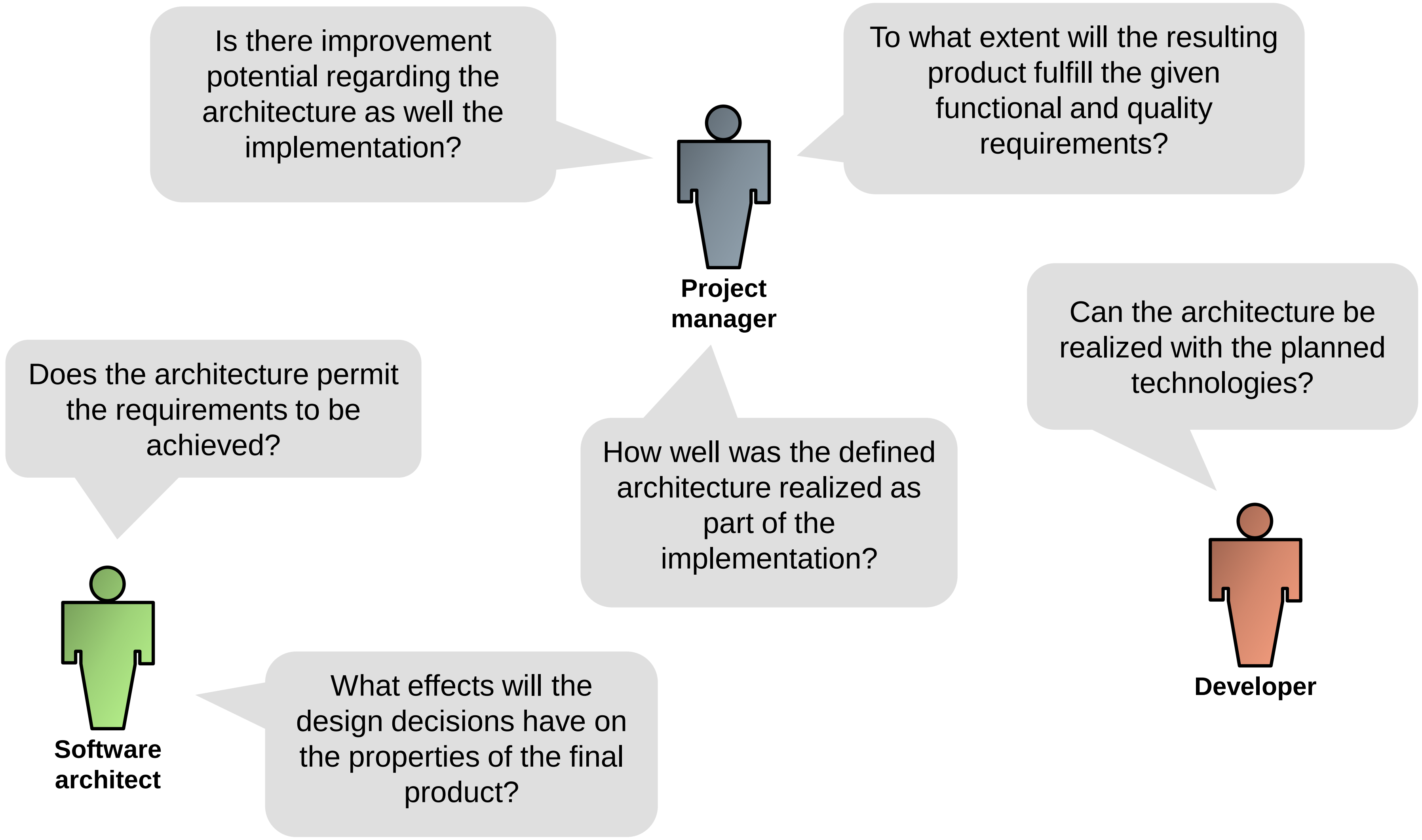
Fraunhofer IESE

Stakeholder Concerns - External Business Perspective


Does my subcontractor deliver high quality? Can they really achieve the promises (time/money) they made?
Can we rely on this product to achieve our business goals in three years?
Management
Is it worth to invest into new technology X? What are the advantages and drawbacks for our products?
What unique selling points does the architecture bring?
What is the best starting point for our new products?
Procurement
Sales and Marketing
What features will the final product have and what qualities will it provide?

© Fraunhofer IESE

Stakeholder Concerns - Internal Technical Perspective



Is there improvement potential regarding the architecture as well the implementation?

To what extent will the resulting product fulfill the given functional and quality requirements?

Can the architecture be realized with the planned technologies?

Does the architecture permit the requirements to be achieved?

How well was the defined architecture realized as part of the implementation?

What effects will the design decisions have on the properties of the final product?

© Fraunhofer IESE

How the customer explained it

What Drives my Architecture?

- Whatever is…
  - Costly to change
  - Risky
  - New

© Fraunhofer IESE

Fraunhofer IESE

Architectural Drivers

- **Business goals**
  - Customer organization
  - Developing organization
- **Quality attributes**
  - System in use (runtime quality attributes)
  - System under development (devtime quality attributes)
- **Key functional requirements**
  - Unique properties
  - Make system viable
- **Constraints**
  - Organizational, legal, and technical
  - Cost and time

© Fraunhofer IESE

Fraunhofer IESE

Notations for Architecture Drivers

Business Goals	Constraints	Quality Attributes	Key Functional Requirements
• **Natural Language** • **Links to Other Documents**	• **Natural Language** • Links to Other Documents	• **Drivers** • **Scenarios** • Links to Other Documents • (Use Cases)	• **Use Cases** • **User Stories / Epics** • Driver • Scenario • Natural Language • Links to Other Documents

© Fraunhofer IESE

Why invest into Architecture Drivers…
if there are so good requirements …?

- Requirements often…
  - are not well analyzed and documented
  - are not complete
  - do not cover development and operation aspects
- Sometimes, amount of requirements is so big that architects have to condense

© Fraunhofer IESE

Problems with Quality Attributes

- **There is no standard set of quality attributes**
  - Maintainability/modifiability/portability
  - People invent new ones…
  - There is no standard meaning of what being secure is
- **How can we measure the achievement of the quality attributes?**
  - Architecture drivers help us to avoid these problems!
  - The quality attributes are defined by the concise drivers!
- **Architecture drivers**
  - Are a central artifact in architecture design and evaluation
  - Are a notation for architecture drivers
  - Allow the precise description of these requirements

© Fraunhofer IESE

Architecture Driver Template

Categorization	
Driver Name	*Concise short name*
Driver ID	*Unique identifier*
Status	*[Open, Elicited, Under Design, Designed, Under Realization, Realized, Done]*
Priority	*[High - Medium – Low]*

Responsibilities	
Supporter	*Stakeholders supporting the driver*
Sponsor	*Stakeholders paying for the driver*
Author	*Responsible for filling this template*
Inspector	*Stakeholders reviewing this driver*

Description		**Quantification**
Environment	*Context and/or initial situation applying to this driver*	▪ *Measurable effects applying to the environment*
Stimulus	*The event, trigger or condition arising from this driver*	▪ *Measurable effects applying to the stimulus*
Response	*The expected reaction of the system to the driver event (black box view putting no constraints on the design)*	▪ *Measurable effects applying to the response* ▪ *Measurable indicators that the driver has been achieved by the architecture*

© Fraunhofer IESE

Architecture Driver Example

Categorization	
Driver Name	Application startup time
Driver ID	AD.01.PERFORMANCE
Status	Realized
Priority	High

Responsibilities	
Supporter	Carla Customer
Sponsor	Mike Manager
Author	Arnold Architect
Inspector	Alfred Architect

Description		Quantification
Environment	The application is installed on the system and has been started before at least once. The application is currently closed and the system is running on normal load.	▪ Previous starts >= 1
Stimulus	A user starts the application from the Windows start menu.	
Response	The application starts and is ready for inputting search data in less than 1 second. The application is ready for fast answers to search queries after 5 seconds.	▪ Initial startup time < 1s ▪ Full startup time < 5s

© Fraunhofer IESE

Fraunhofer IESE

Architecture Driver Example
Most Important

Categorization	
Driver Name	Application startup time
Driver ID	AD.01.PERFORMANCE
Status	Realized
Priority	High

Responsibilities	
Supporter	
Sponsor	
Author	
Inspector	

Description		Quantification
Environment	The application is installed on the system and has been started before at least once. The application is currently closed and the system is running on normal load.	▪ Previous starts >= 1
Stimulus	A user starts the application from the Windows start menu.	
Response	The application starts and is ready for inputting search data in less than 1 second. The application is ready for fast answers to search queries after 5 seconds.	▪ Initial startup time < 1s ▪ Full startup time < 5s

© Fraunhofer IESE

Fraunhofer IESE

Architectural Drivers – Examples

- „A user wants to update the system. The update is triggered with a maximum of 3 clicks. "
- „During operation, a server fails. All ongoing operations are unaffected by the failure"
- „Each user input generates a visual response within 0.2 s"
- „A new feature is to be implemented. A team of 5 people is able to realize the feature within three days"
- „We are not allowed to use Open Source software at all"
- „We want to change our complete business model to SaaS"

AD3: „Under high system load due to background processing of computation-intensive operations, each user input in the GUI is processed within 0.2 s"

© Fraunhofer IESE

Fraunhofer IESE

Quantification and Measures can be expressed relatively!

Environment	#msgs in Inbox
Stimulus	Start Outlook
Response	Outlook shows the inbox
Response Measure	within #msgs/10 [ms]

Environment	System is idle
Stimulus	System gets X requests per second
Response	System reacts
Response Measure	after X / 10s

© Fraunhofer IESE

Fraunhofer IESE

Purpose of Architectural Drivers

- **Compensation**
  - of missing (unknown) requirements
  - of complex exceptional cases
- **Aggregation**
  - of large amounts of similar (types of) or repeating requirements
- **Consolidation**
  - of different stakeholder opinions and concerns (business vs. technical)
  - of investments into future (anticipate change)
- **Negotiation**
  - between external quality (runtime) and internal quality (devtime)
  - to align conflicting stakeholder concerns
  - to meet constraints

© Fraunhofer IESE

Fraunhofer IESE

Compensation of Architectural Drivers

What we typically find in practice as architects

- *Business goals*
  - often found, but not well understood
- *Functional requirements*
  - often found
- *Runtime quality attributes*
  - often found, but not specific enough
- *Devtime quality attributes*
  - rarely found, seldom specific
- *Operation quality attributes*
  - rarely found
- *Constraints*
  - often found, but not always really fix

→ Architects have spend work for compensation of architectural drivers

© Fraunhofer IESE

QAs Important for Distributed Systems

- **Availability**: ability to continue operation after a computer/piece of equipment failed
- **Reliability**: continuity of correct service
- **Performance**: timely response to service request events, throughput, jitter
- **Scalability**: continue to function as expected when it (or its context) is changed in size or volume
- **Security**: ability to resist unauthorized attempts to access data and services
- **Safety**: ability to mitigate consequences of critical failures
- **Integrity**: absence of improper system alterations

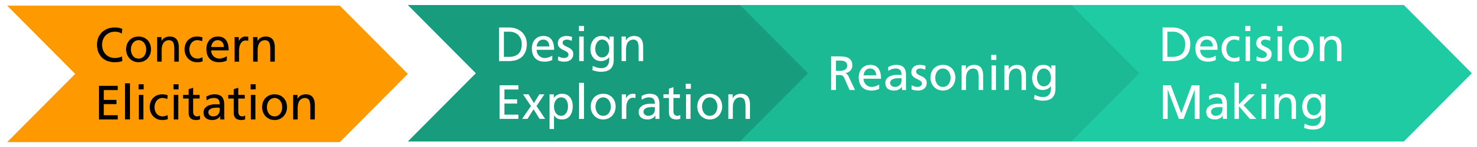
- **Openness**: use of equipment and software from different vendors
- **Maintainability**: ability to undergo modifications and repairs
- **Testability**: verification of the correctness of the system
- **Portability**: ability to port system to other platforms / technologies

© Fraunhofer IESE

Fraunhofer IESE

Driver Solutions

From Drivers to Solutions



© Fraunhofer IESE

Driver Solution Template

Driver Name	*Concise short name*
Driver ID	*Unique identifier*

Steps	*1. Logical flow to explain driver solution (white box view explaining the design)* *2. The glue between design decisions (accepted and discarded)* *3. Putting all related design decisions in a combined and larger context*	
Related Design Decisions	*ACCEPTED* ▪ *Link to design decision (detailed description) to enable traceability*	*DISCARDED* ▪ *Link to design decision (detailed description) to enable traceability*

Pros & Opportunities	**Cons & Risks**
▪ *Points in favor* ▪ *Anticipations of future*	▪ *Points against* ▪ *Unknown or open aspects*

Assumptions & Quantifications	**Trade-Offs**
▪ *Assumption made about the driver solution (or parts of it)* ▪ *Measurable effects applying to the driver solution (or parts of it)*	▪ *Trade-offs to other design decisions, quality attributes, solutions concepts, architecture drivers* ▪ *Potentially impacted if this solution changes*

Manifestation Links	*Links to models, diagrams, additional documentation*

© Fraunhofer IESE

Fraunhofer IESE

Driver Solution Example

Driver Name	Application startup time
Driver ID	AD.01.PERFORMANCE.

Steps	1. Application always stores preprocessed index-structures on updates of searchable items 2. On startup, loading of search data is moved to a separate thread 3. The UI is started and ready for user input while loading of search data is ongoing 4. After loading the search data, searches can be done without the user noticing that search was not available before
Related Design Decisions	▪ DD.01 Decoupled loading of search data ▪ DD.12 Preprocessed index-structures of search data

Pros & Opportunities	**Cons & Risks**
▪ Very fast startup time, application directly usable by user	▪ More effort in realization ▪ Loading in separate thread requires synchronization and makes implementation more difficult

Assumptions & Quantifications	**Trade-Offs**
▪ Data can be loaded in 5s ▪ User rarely sends a search in less than 4s after start is completed	▪ Maintainability, understandability

© Fraunhofer IESE

Fraunhofer IESE

Decision Rationale Template

Decision Name	*Concise short name*
Design Decision ID	*Unique identifier*
Explanation	*Explanation of the decision rationale*

Pros & Opportunities

- *Points in favor*
- *Anticipations of future*

Cons & Risks

- *Points against*
- *Unknown or open aspects*

Assumptions & Quantifications

- *Assumption made about the driver solution (or parts of it)*
- *Measurable effects applying to the driver solution (or parts of it)*

Trade-Offs

- *Trade-offs to other design decisions, quality attributes, solutions concepts, architecture drivers*
- *Potentially impacted if this solution changes*

Manifestation Links	*Links to models, diagrams, additional documentation*

© Fraunhofer IESE

Decision Rationale Example

Decision Name	Decoupled loading of search data
Design Decision ID	DD.01
Explanation	Loading the search data is done in a separate thread. The application's UI can be started and used for typing in search queries before the search data is actually loaded.

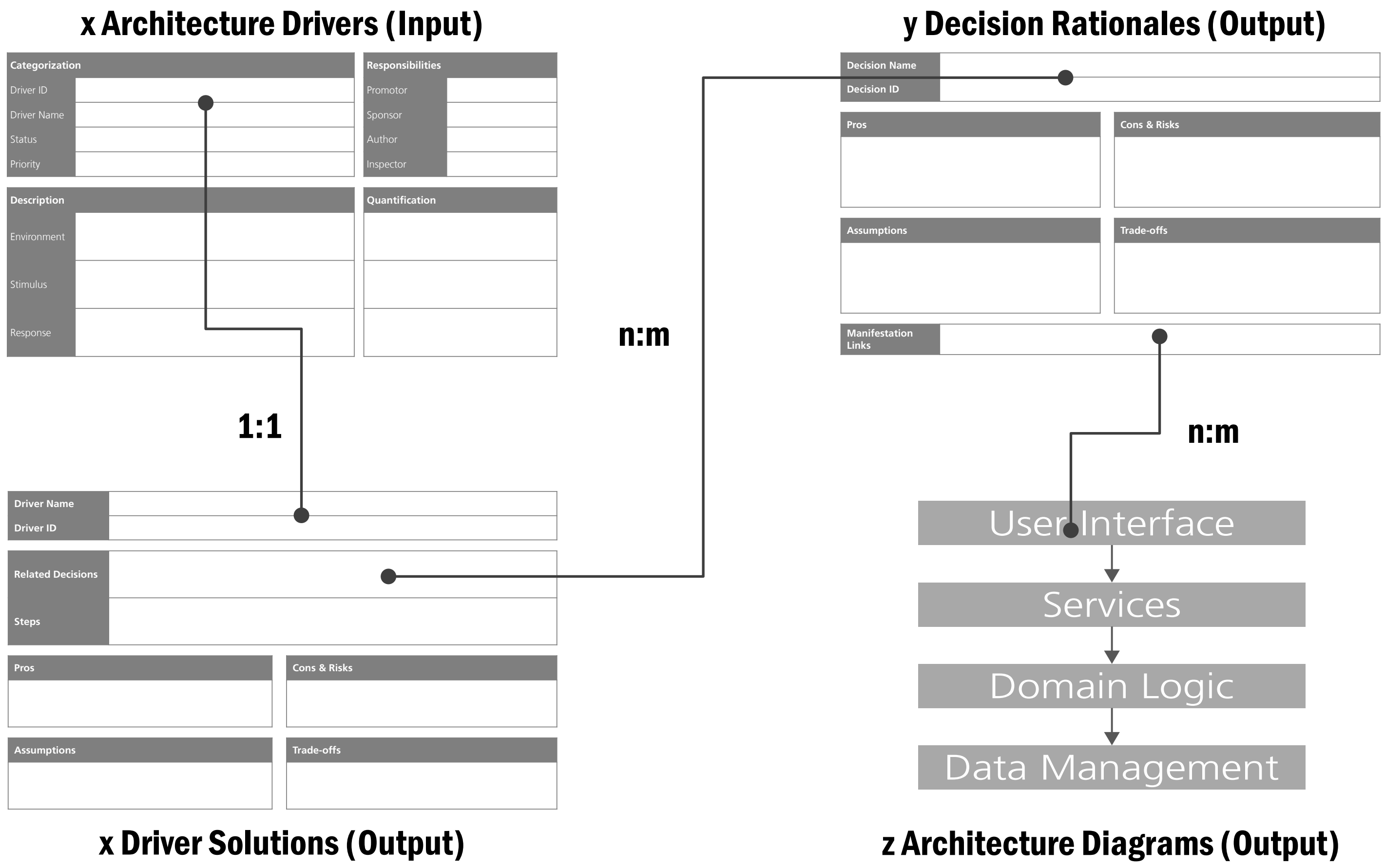
Pros & Opportunities	**Cons & Risks**
▪ Data loading time does not add on startup time	▪ Loading in separate thread requires synchronization and makes implementation more difficult

Assumptions & Quantifications	**Trade-Offs**
▪ Data can be loaded in 5s	▪ Maintainability, understandability

Manifestation Links	

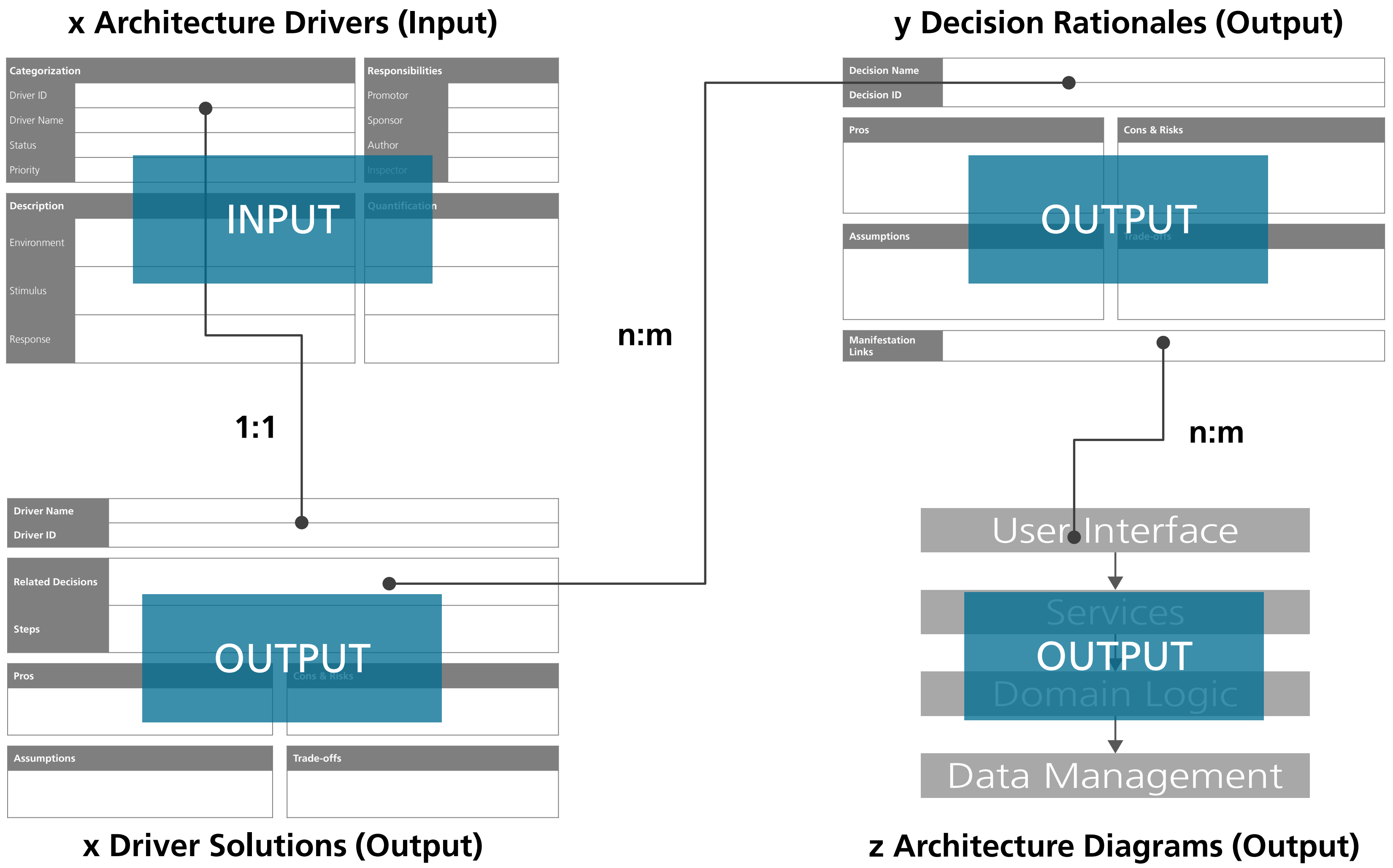
© Fraunhofer IESE

Fraunhofer IESE



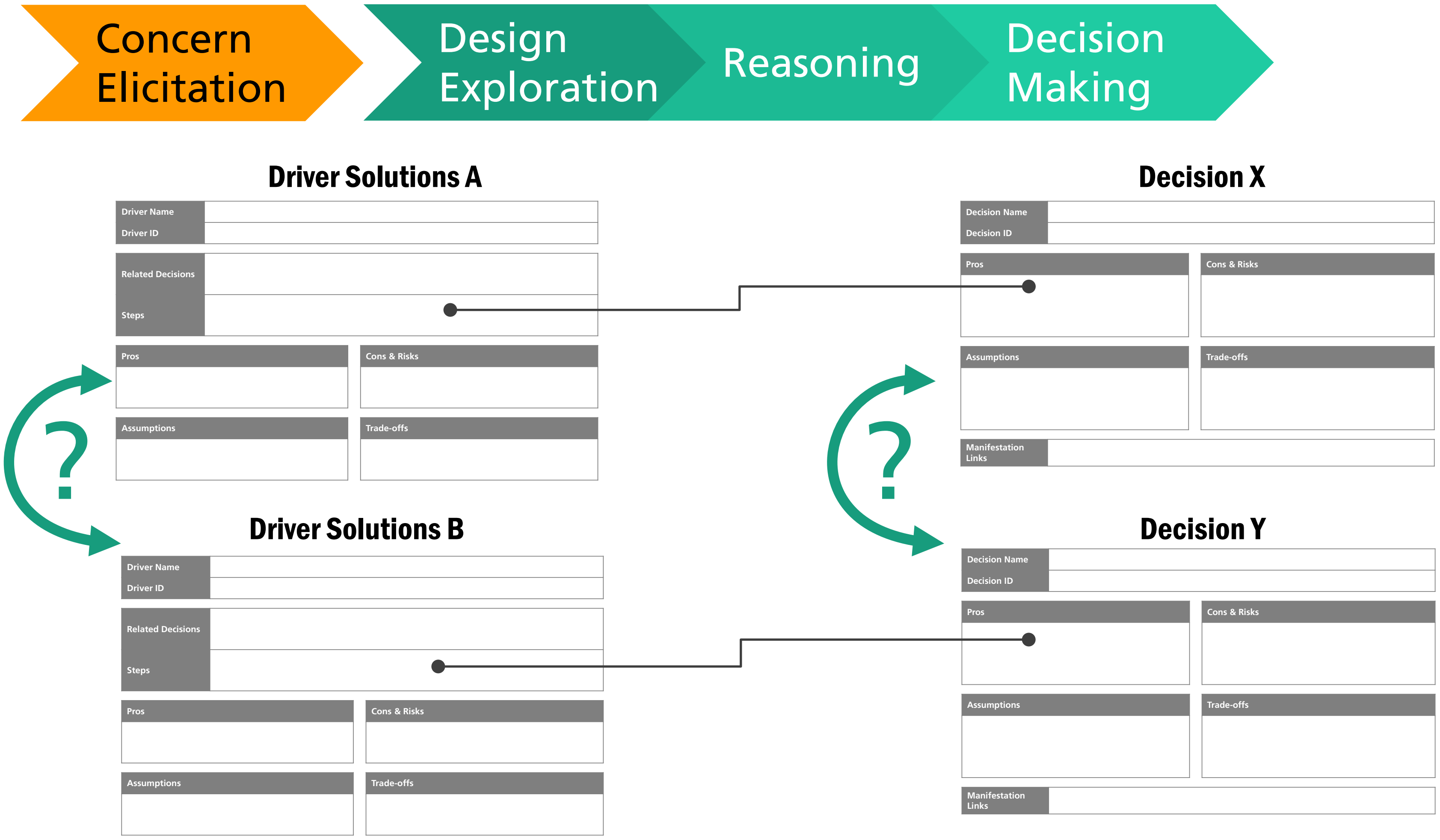
© Fraunhofer IESE

Fraunhofer IESE



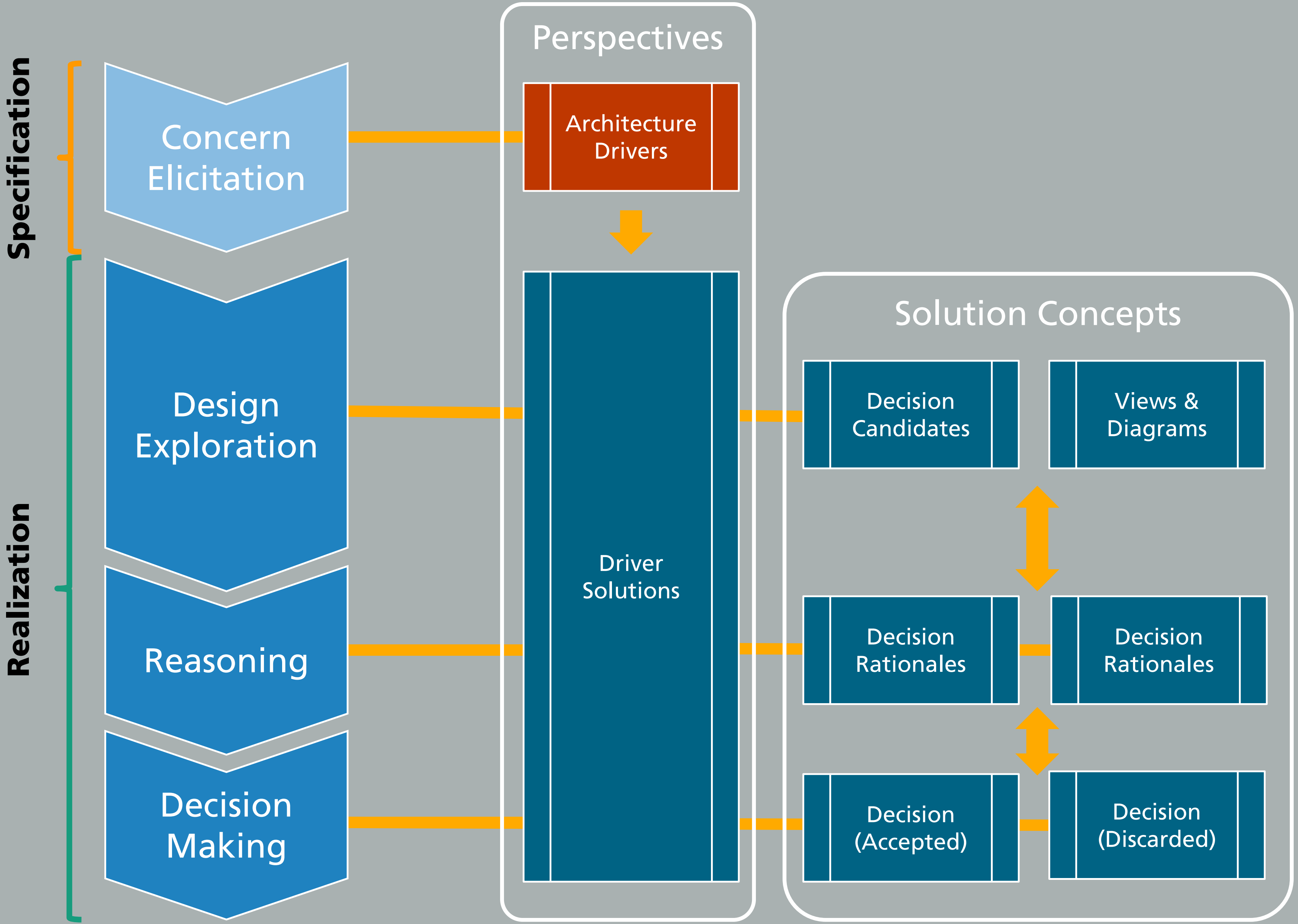
© Fraunhofer IESE

From Drivers to Solutions



© Fraunhofer IESE

Architecture Decision Making



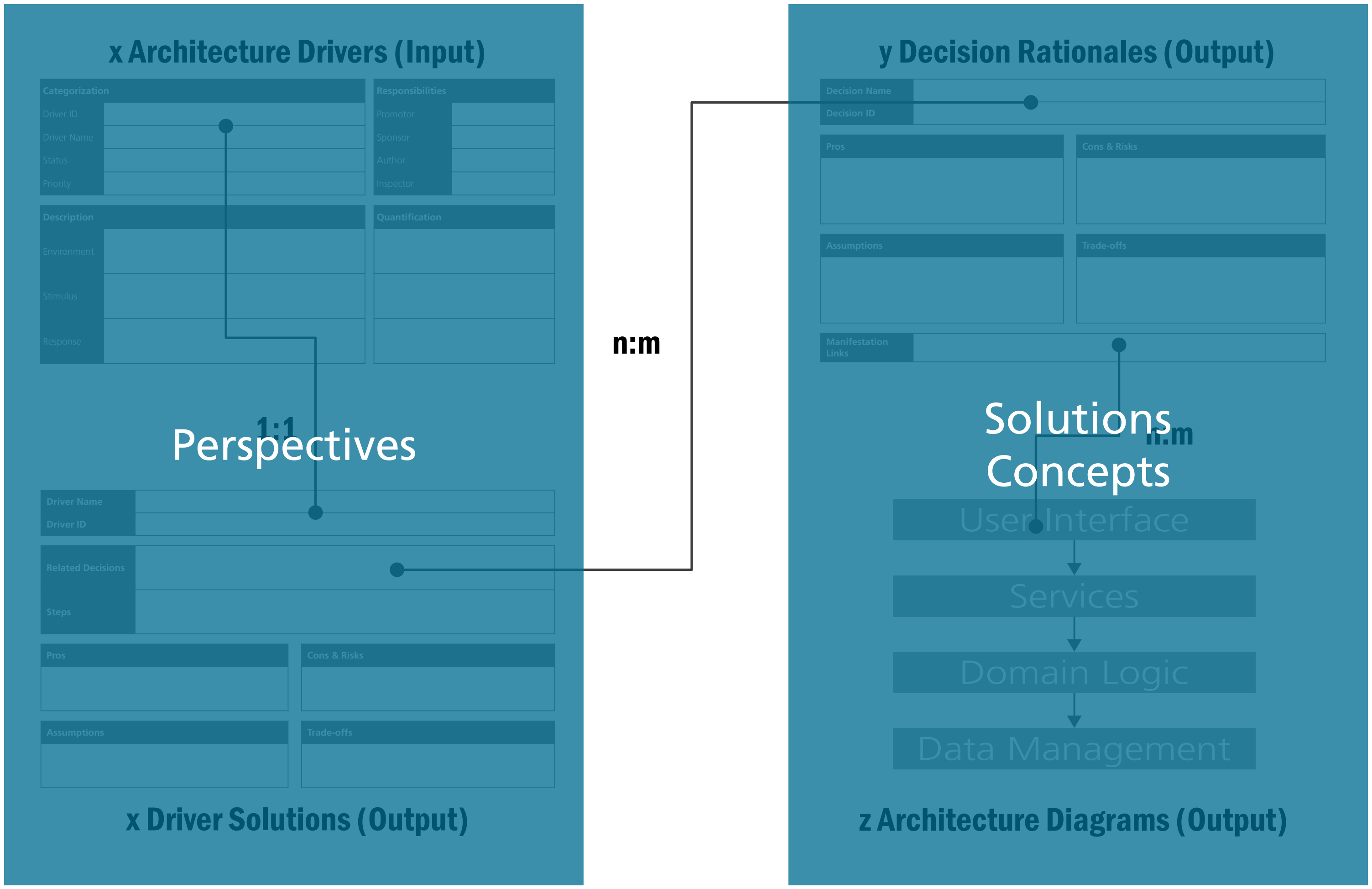
Perspectives

- **Perspectives** organize the modeled trace between
  - Driver
  - Driver Solution
    - Solution Concepts
    - Decision Rationale

- Shaping a perspective …
  - … along architectural drivers
    → Put focus only on **related steps and design decisions**
  - … on **relevant solution parts**
    → Add navigation links to relevant solution concepts (architectural views)

© Fraunhofer IESE

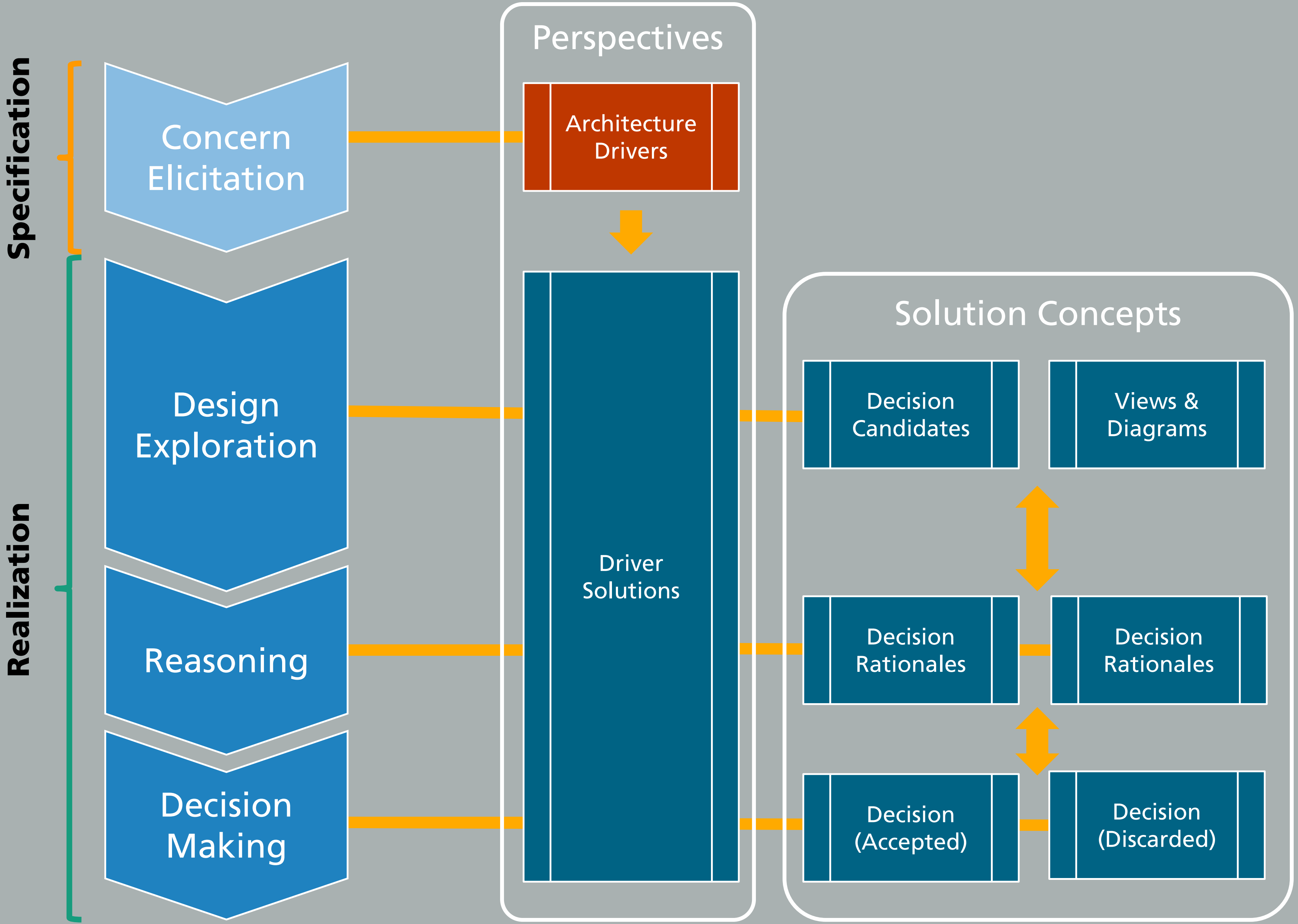
Fraunhofer IESE



© Fraunhofer IESE

Wrap Up

Architecture Decision Making



Architectural Drivers

- **Business goals**
  - Customer organization
  - Developing organization
- **Quality attributes**
  - System in use (runtime quality attributes)
  - System under development (devtime quality attributes)
- **Key functional requirements**
  - Unique properties
  - Make system viable
- **Constraints**
  - Organizational, legal, and technical
  - Cost and time

© Fraunhofer IESE

Fraunhofer IESE

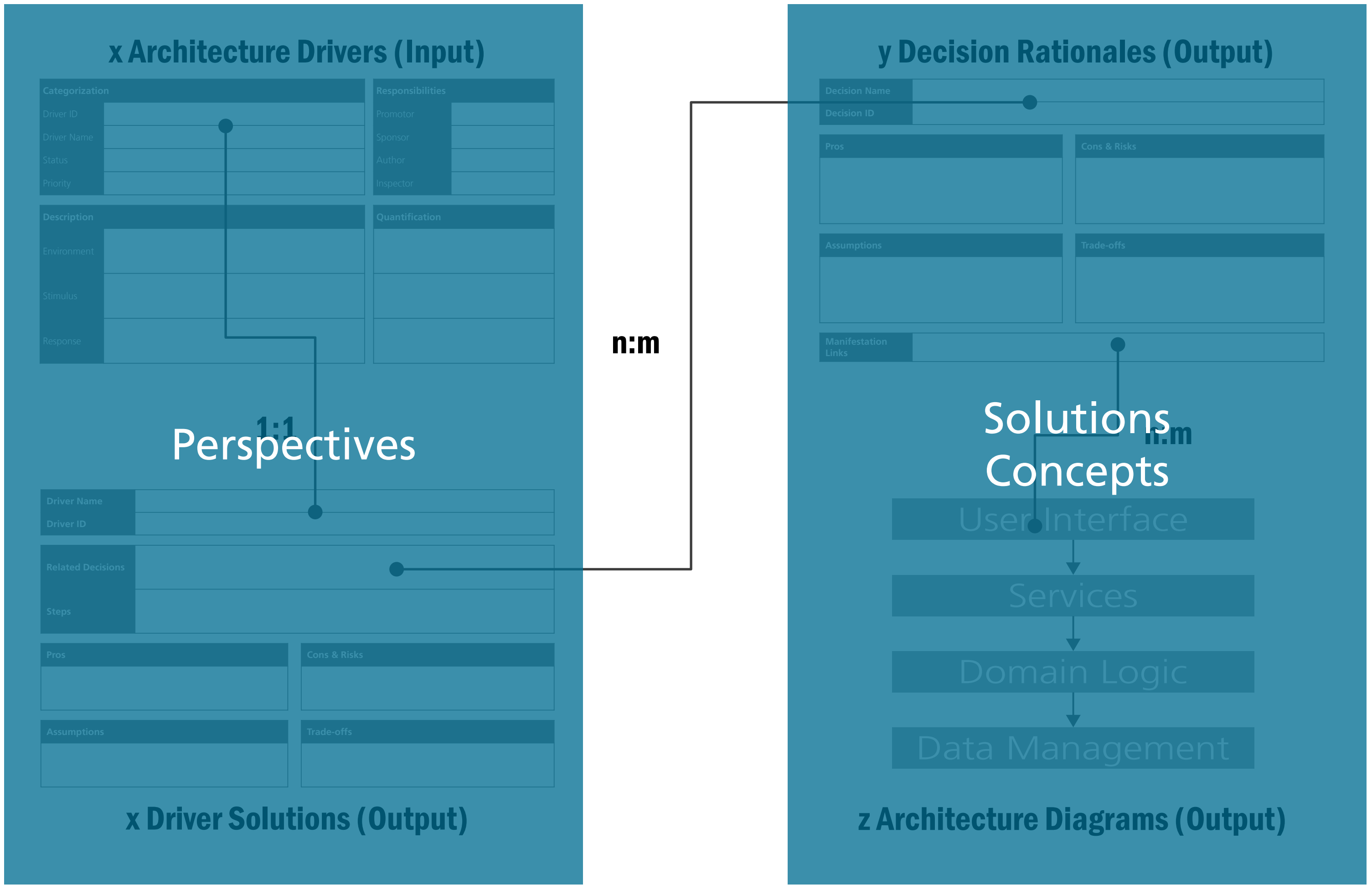
QAs Important for Distributed Systems

- **Availability**: ability to continue operation after a computer/piece of equipment failed
- **Reliability**: continuity of correct service
- **Performance**: timely response to service request events, throughput, jitter
- **Scalability**: continue to function as expected when it (or its context) is changed in size or volume
- **Security**: ability to resist unauthorized attempts to access data and services
- **Safety**: ability to mitigate consequences of critical failures
- **Integrity**: absence of improper system alterations

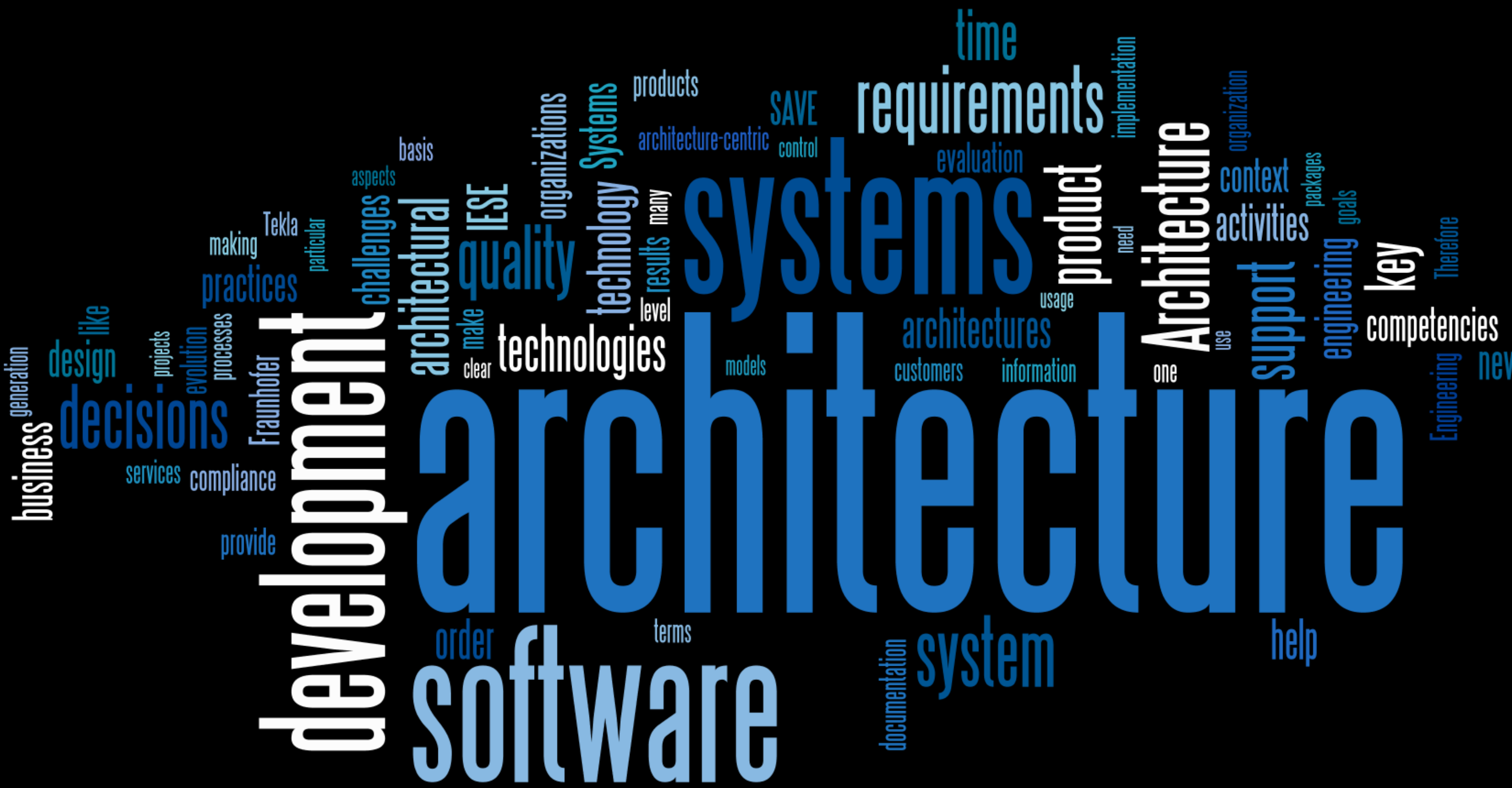
- **Openness**: use of equipment and software from different vendors
- **Maintainability**: ability to undergo modifications and repairs
- **Testability**: verification of the correctness of the system
- **Portability**: ability to port system to other platforms / technologies

© Fraunhofer IESE

Fraunhofer IESE



© Fraunhofer IESE


time
products
requirements
implementation
SAVE
organization
architecture-centric
control
basis
organizations
Systems
evaluation
aspects
context
packages
goals
Tekla
particular
challenges
architectural
IESE
technology
many
systems
product
Architecture
activities
making
quality
need
key
Therefore
practices
results
level
usage
support
engineering
like
design
projects
evolution
processes
make
technologies
architectures
use
competencies
generation
Fraunhofer
development
clear
models
customers
information
one
nev
decisions
Engineering
business
architecture
services
compliance
provide
terms
order
help
software
documentation
system