

SSA 01 – Hello Architecture

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

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



Hello Architecture

Discussion



- What is software architecture?
 - State an “intuitive” definition of the term

- Why having a software architecture?
 - Your experiences?
 - Your challenges?
 - Your solutions?

- What does an architect do?
 - Why is architecting needed / useful?
 - Role in software engineering?
 - Skills and expertise architects need?

What is Architecture?[1]

- Modules, connections, dependencies and interfaces
- „The big picture“
- An abstraction
- Things that are expensive to change
- A conceptual model
- Satisfying non-functional reqs /quality attributes
- A plan
- A blueprint
- Systems, subsystems, interactions, and interfaces
- Governance
- The outcome of strategic decisions
- Necessary constraints
- Tools and methods
- Technical leadership
- Strategy and vision
- ...

Software Architecture Definitions

- Software architecture is the **structure or structures** of the **system**, which comprise software **elements**, the **externally visible properties** of those elements, and the **relationships** among them.

[Software Architecture in Practice, L.Bass, P.Clements, R.Kazman]

- Software architecture is the fundamental **concepts or properties of a system** in its **environment** embodied in its **elements, relationships**, and in the **principles** of its **design and evolution**.

[Systems and software engineering — Architecture description, ISO Standard 42010]

- Software architecture is the **set of design decisions** which, if made incorrectly, may cause your project to be cancelled.

[E. Woods]

- Software architecture is the set of **principal design decisions** made about the **system**.

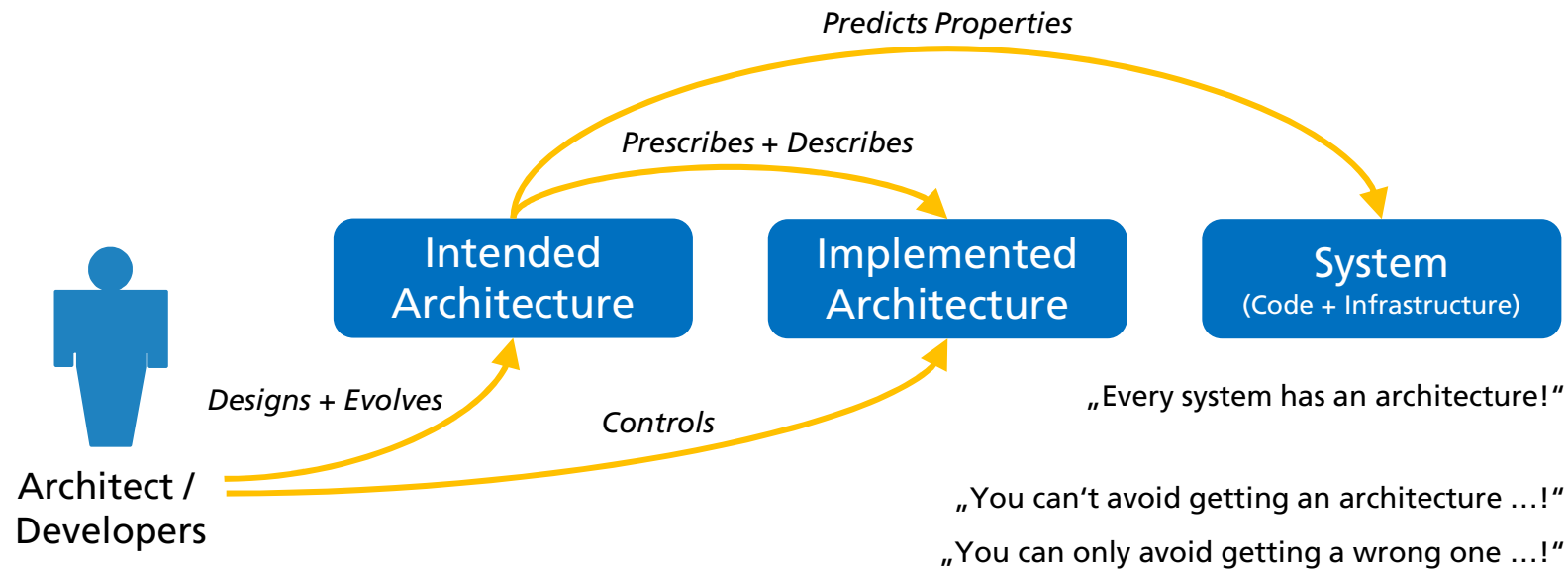
[Software Architecture: Foundations, Theory, and Practice , E.Dashofy, N.Medvidovic, R. Taylor.]

Management Objectives

- **Construction, delivery and maintenance** of **innovative** software systems with **predictable** and adequate **quality** delivered in **time** and **budget**



Foundations of Architecture



„Although the system has an architecture, it might not be known!“

„If it is not clear who makes the decisions, someone will make them!“

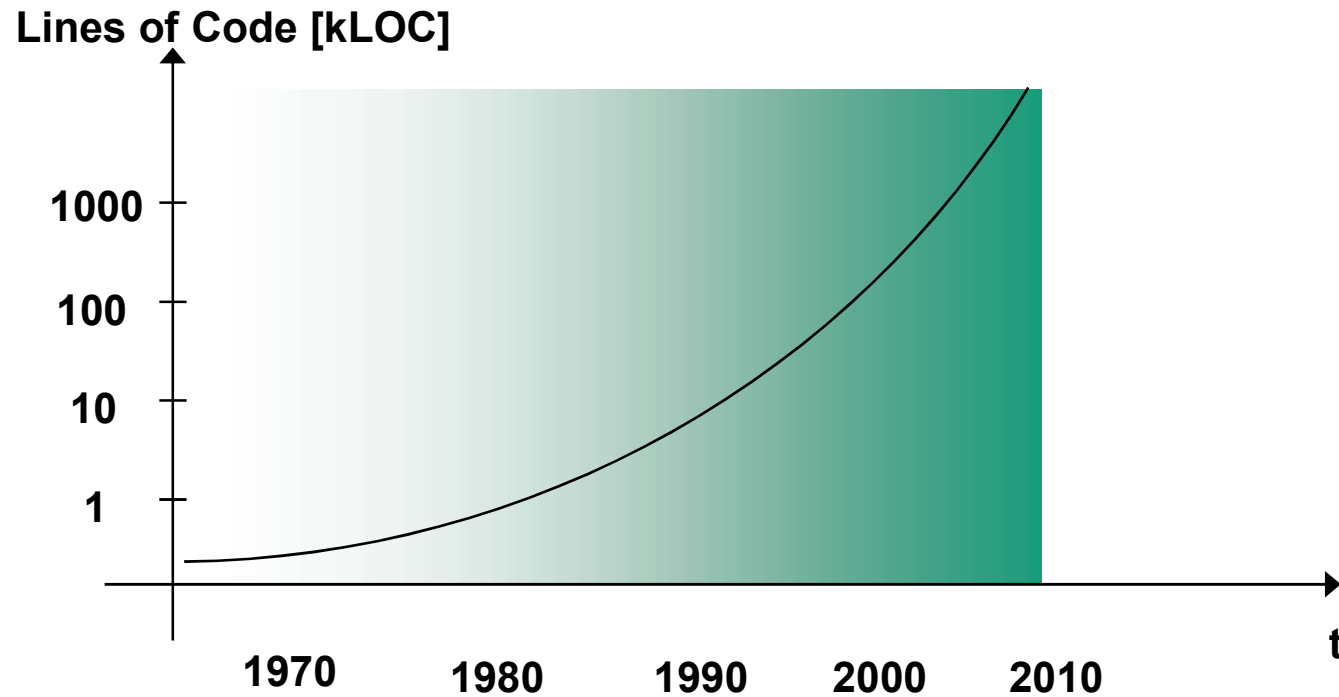
„Architecture is not a phase!“

Challenges in Software Engineering

Challenges and complexity arise from ...

- the products to be built
- the increasing interconnection of systems
- the integration with already existing systems
- the continuous change of systems
- the collaboration of development organizations

Engineering Challenge: Large-Scale Systems



■ Examples

- Car window opener 10.000 LoC
- Car control unit 15.000.000 LoC
- Windows XP 40.000.000 LoC

Engineering Challenge: Large Development Teams

Increasing system size cannot be compensated with more efficient methods

- Large teams have to collaborate
- Teams
 - Distributed over buildings, countries, continents
 - Distributed over departments, organizations
- Decomposition of work for parallelization is essential

Engineering Challenge: High Quality

Quality is not only about **correctness of functionality**

Successful software systems have to assure additional properties

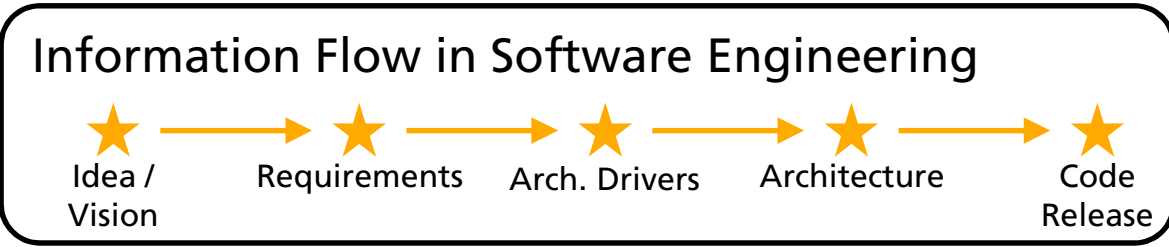
- Performance
- Security
- Availability
- Maintainability
- ...

These properties are the so-called **Quality Attributes**

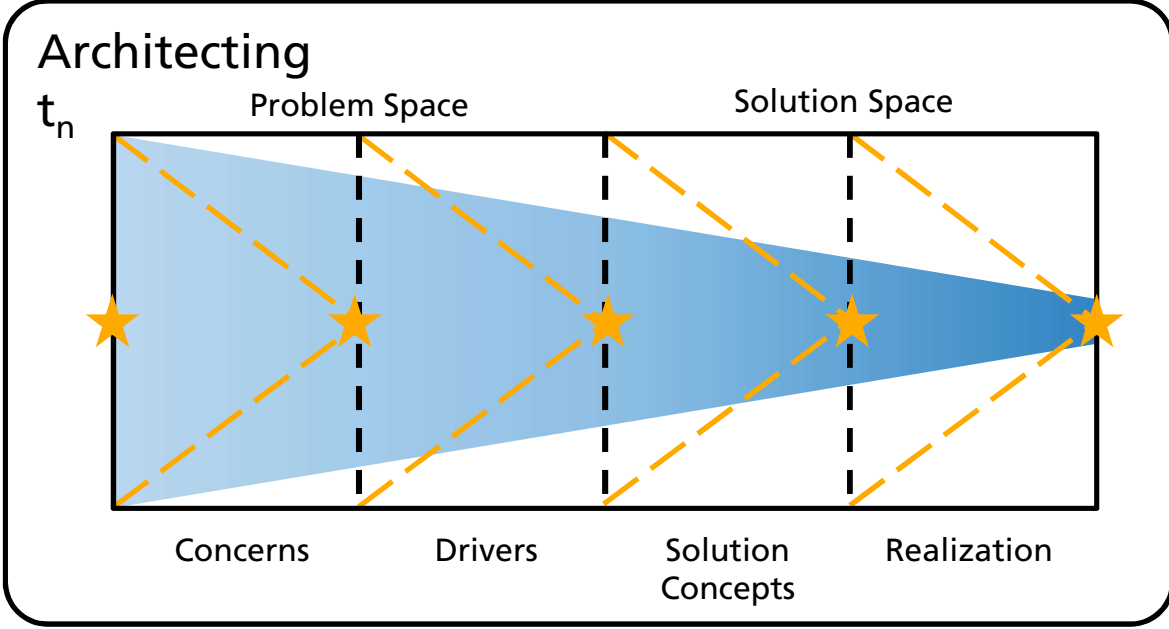
The Mission of Architecture

Conceptual tool to cope with complexity
in Software Engineering needed

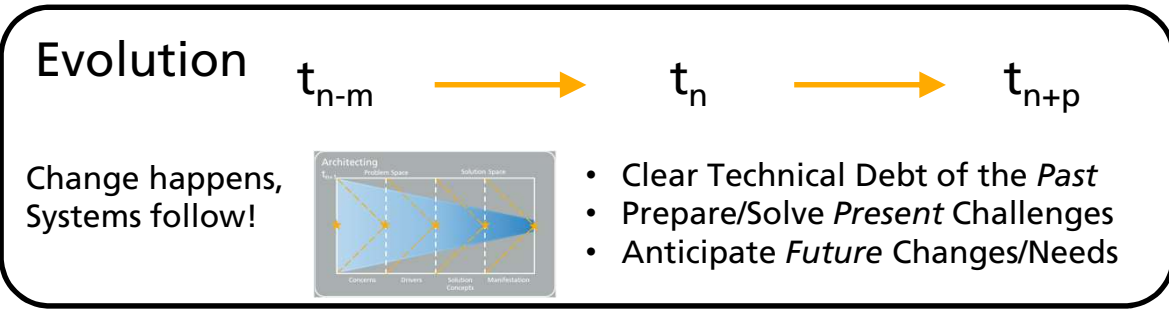
Why Architecture? It is all about Bridging the Gaps!



Concerns
vs.
Results



Problem Space
vs.
Solution Space



Present
vs.
Past & Future

Architecting vs. Architecture

Activities

Design
Modeling
Communication
Negotiation



Artefacts

Design Decisions
Blueprints & Models
Documentation
Implemented Decisions



Architectures: The Artifact

■ ... **provide guidance**

- Plan for constructing a system
- Technical leadership and coordination
- Standards and consistency

■ ... **balance technical risks**

- Identification and mitigation
- Definition of solution concepts
- Anticipation (preparation) for changes

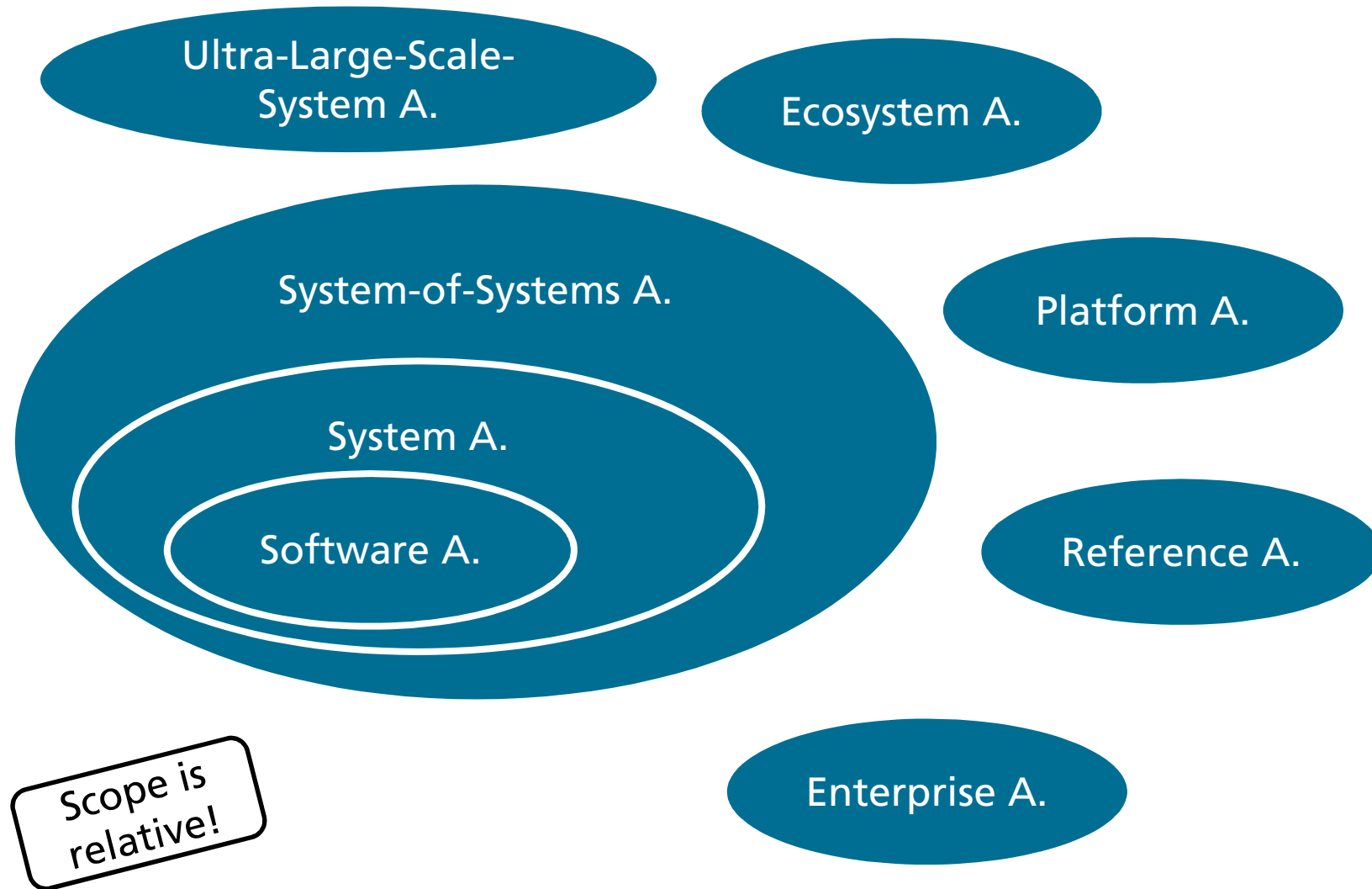
■ ... **enable communication**

- Clear technical vision and roadmap
- Explicit documentation for communication

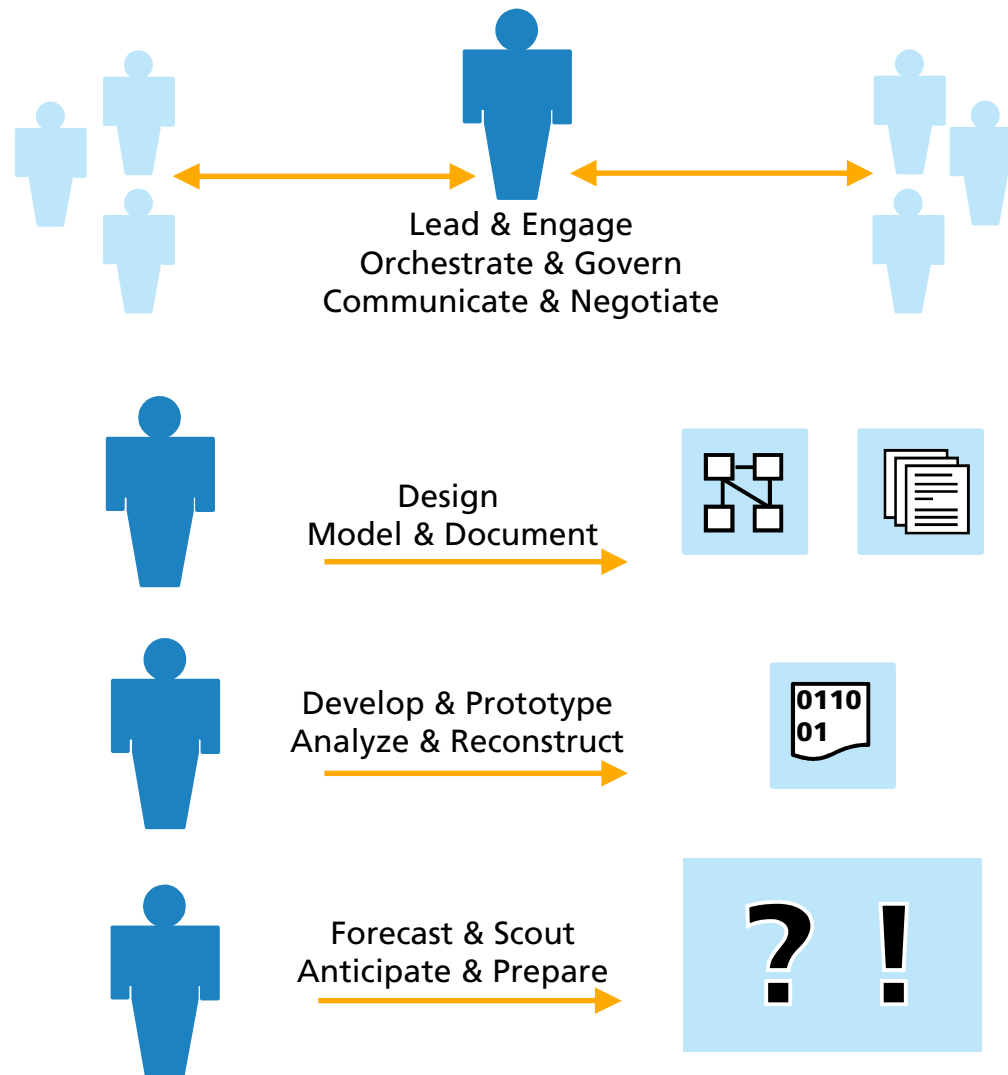
■ ... **manage the inherent complexity** of software

- Products to be built
- Increasing interconnection of systems
- Integration with legacy systems
- Collaboration of organizational units

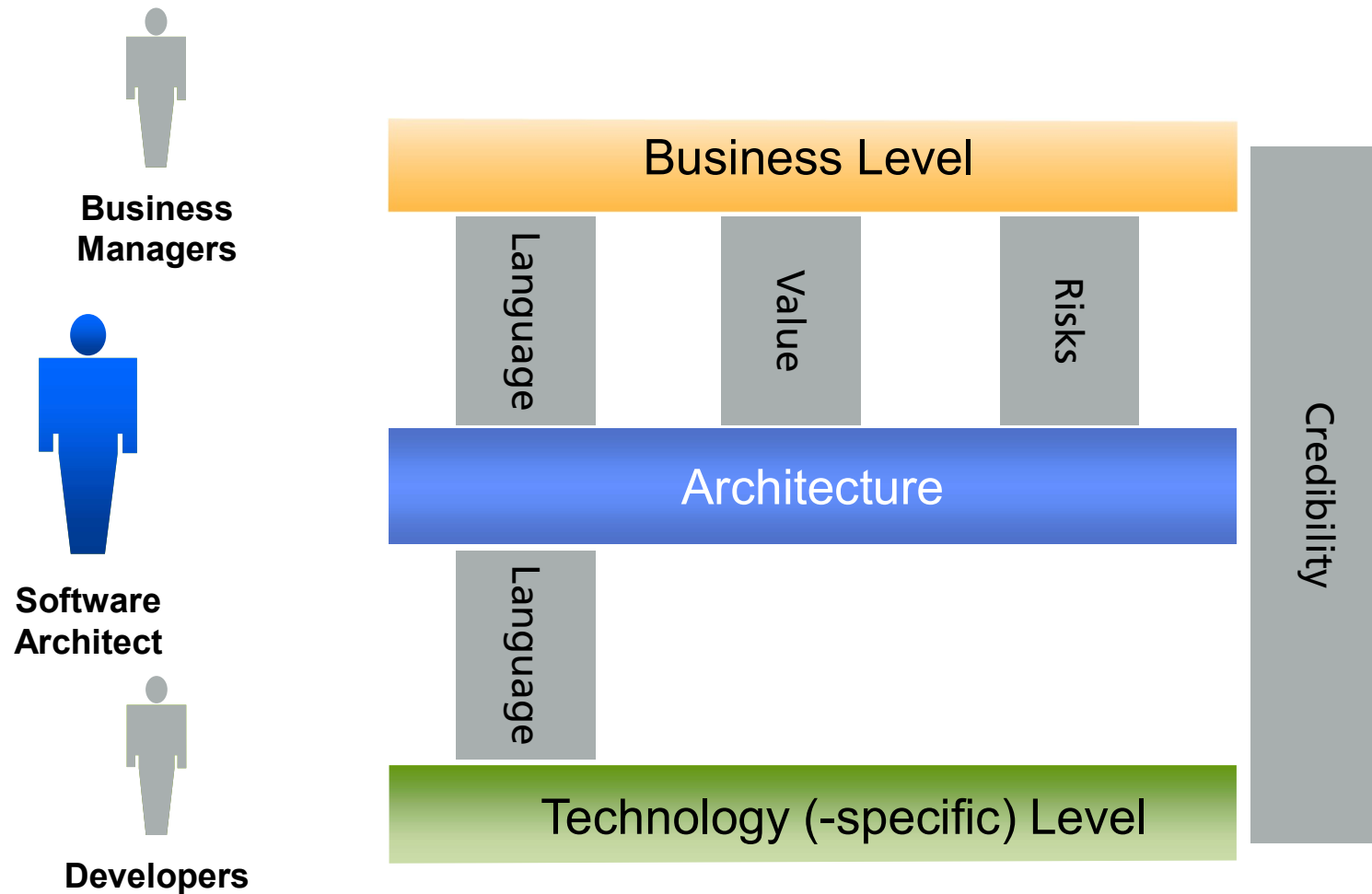
Architecture: It is all about the Scope!



Architecting: The Activity



Architecting: It is all about Speaking the same Language!

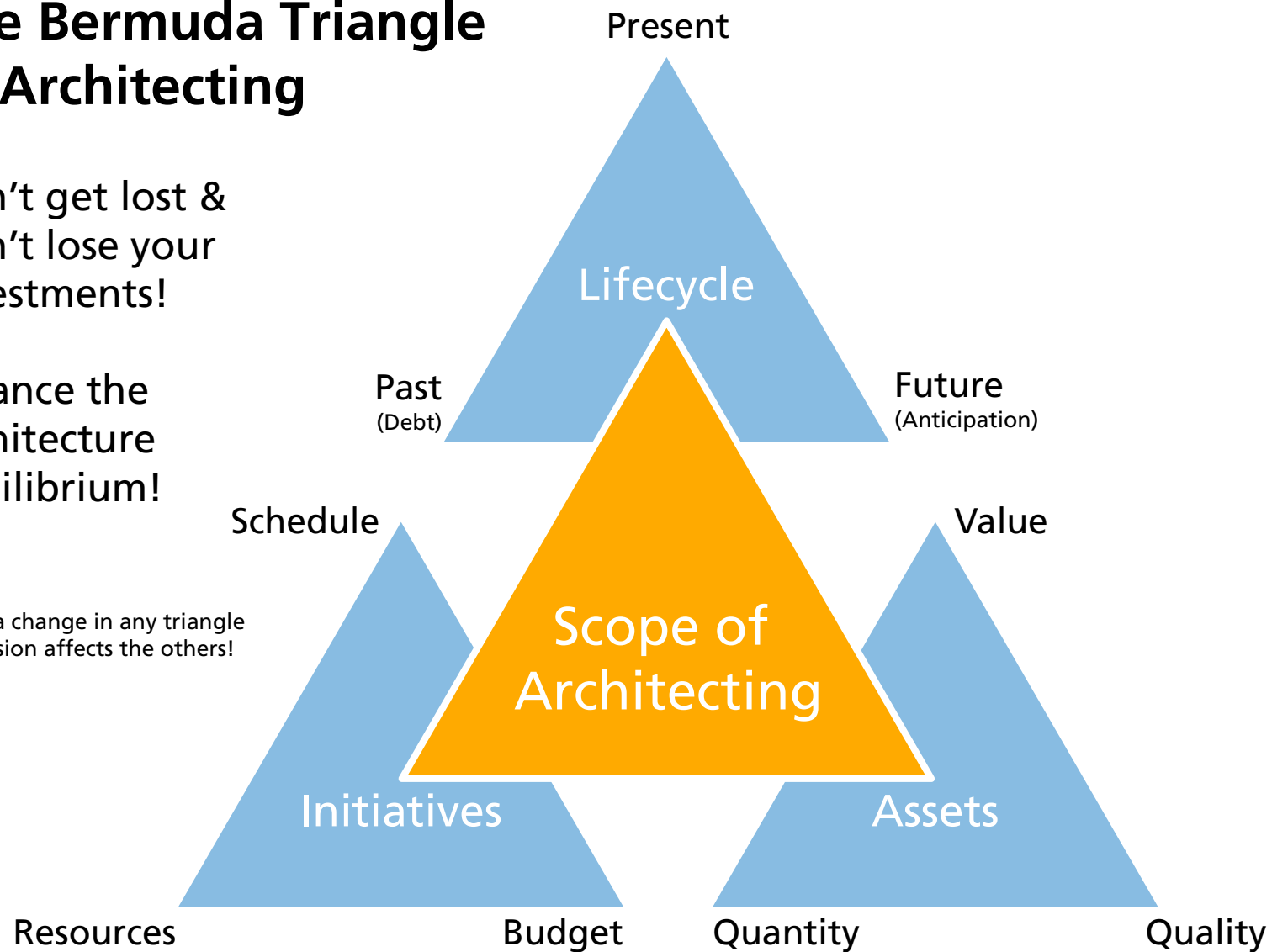


The Bermuda Triangle of Architecting

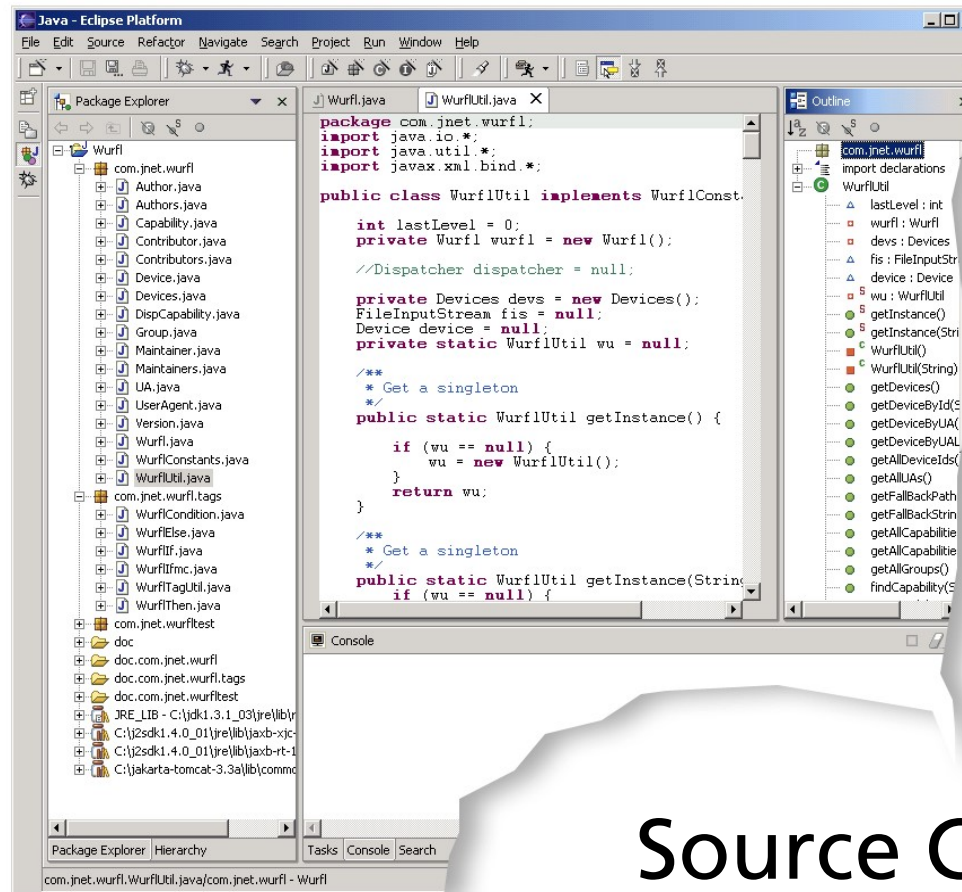
Don't get lost &
Don't lose your
investments!

Balance the
architecture
equilibrium!

Note: a change in any triangle
dimension affects the others!



Real Life: "Source Code is the Only Truth"



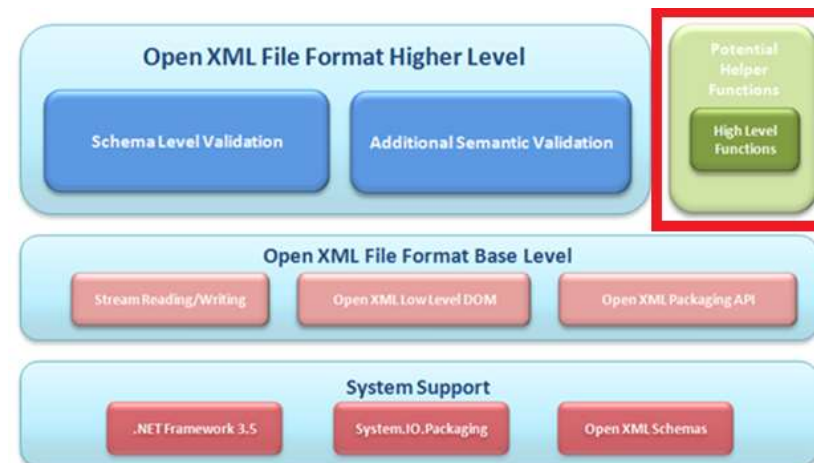
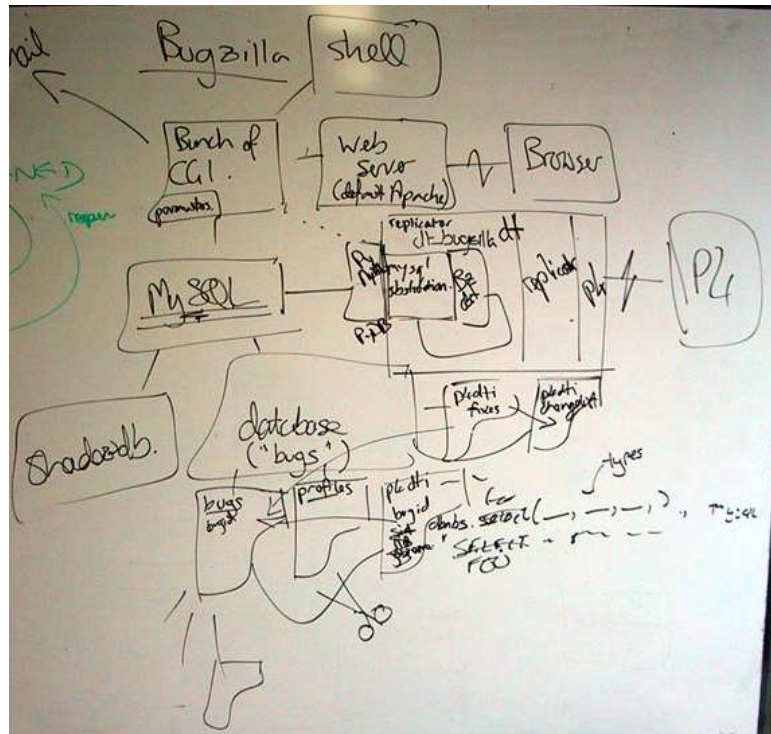
Source Code

Real Life: "I Can Always Explain How the System..."



[Source: dreamstime.com]

Real Life: White Board and PowerPoint Sketches



Real Life: Architecture Documents



Too Long;
Did not Read

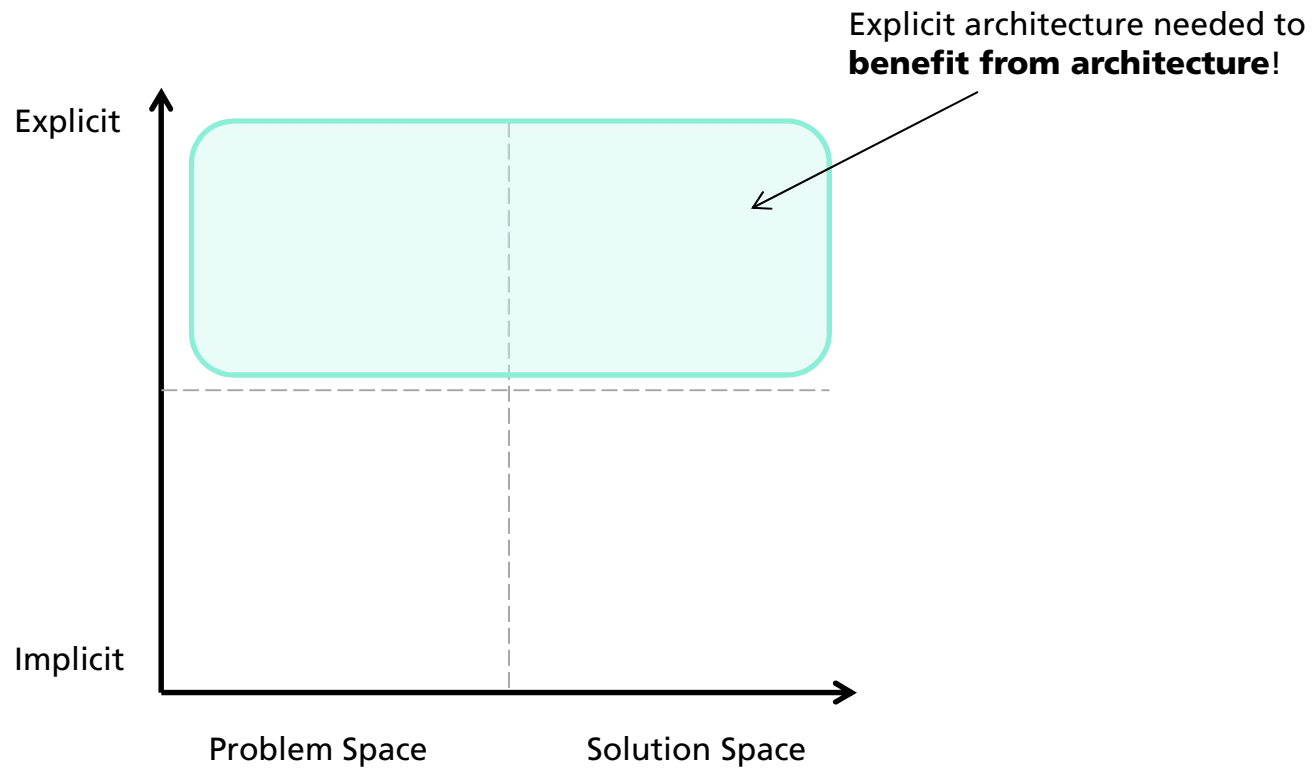
An Ideal Architecture Documentation...

... describes what the code itself does not!

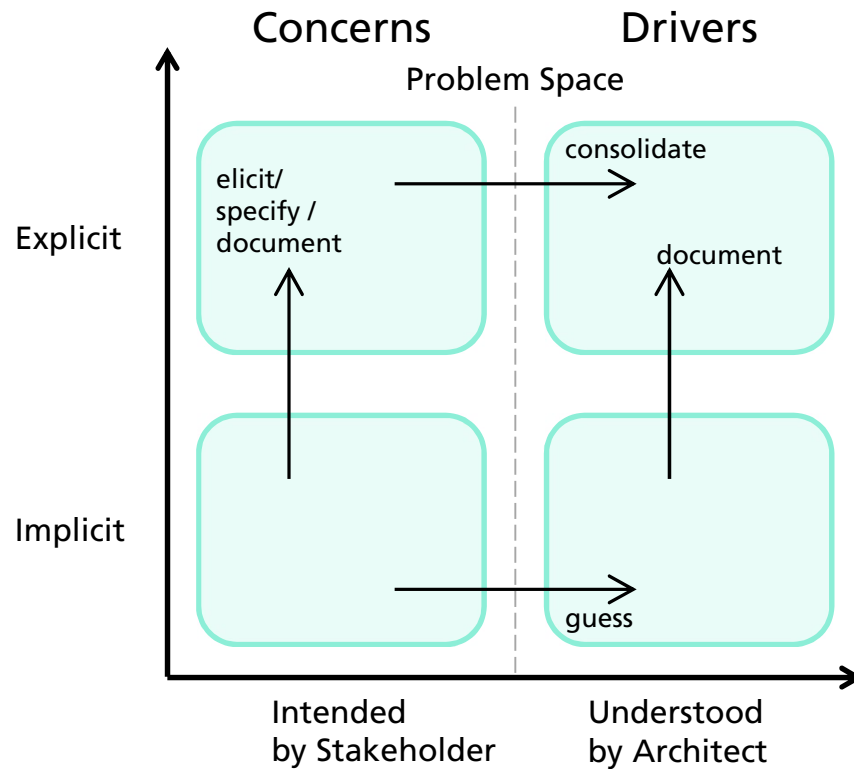
e.g.

- What are the design decisions?
- What is the rationale for the decisions?
- What are the discarded alternatives? Why?
- ...

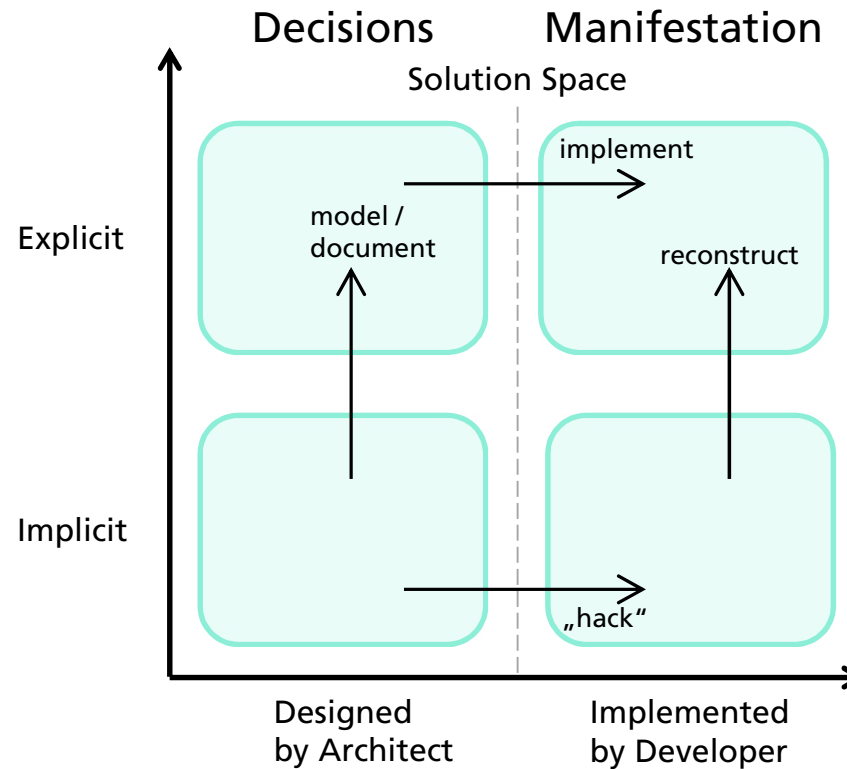
What do We Need in Terms of Architecture?



Explicit vs. Implicit Architecture Problem Space

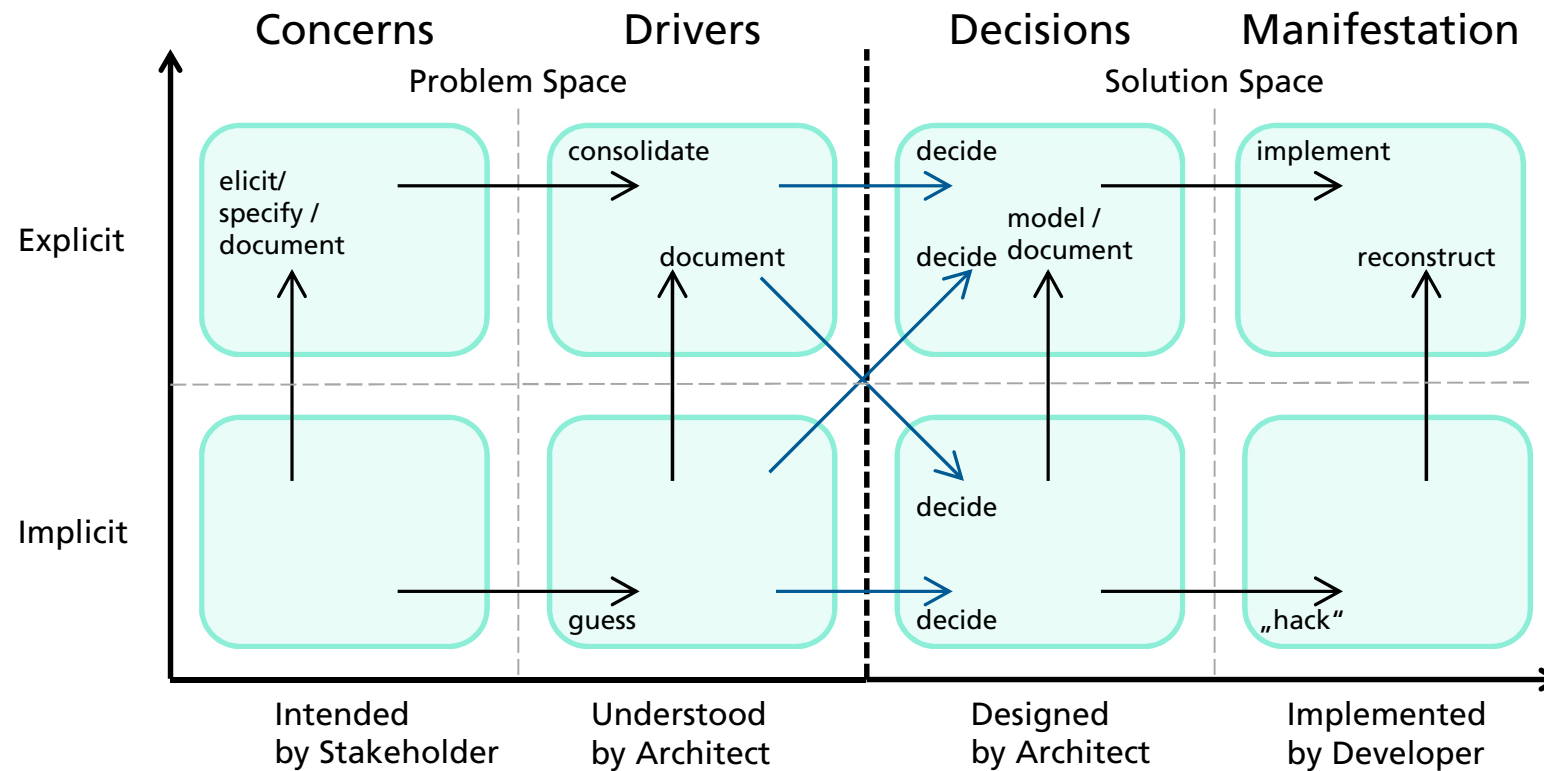


Explicit vs. Implicit Architecture Solution Space

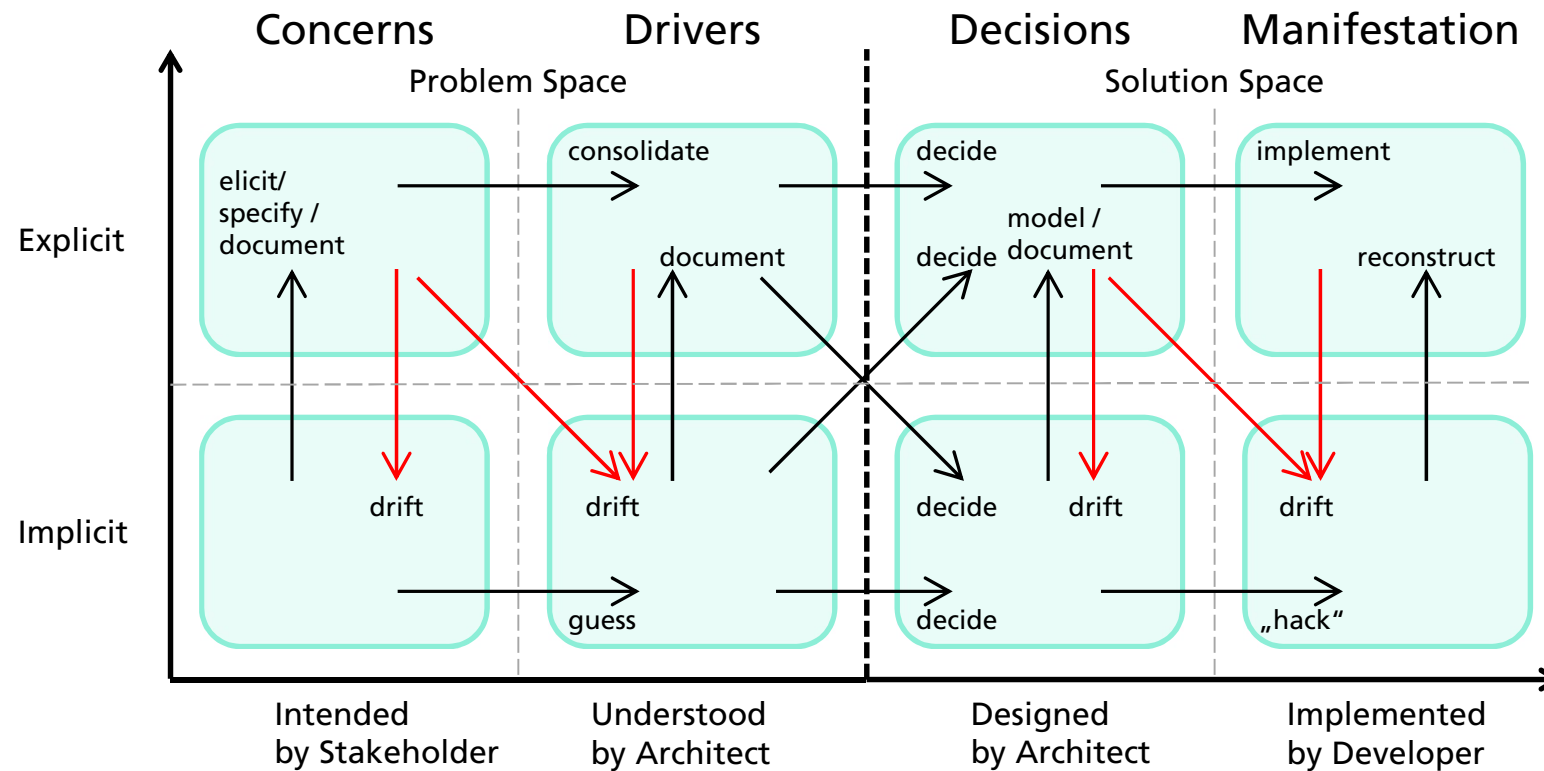


Explicit vs. Implicit Architecture

Problem Space vs. Solution Space



Evolution and Drift



The Architecture of "Hello World"

```
public class HelloWorld {  
    public static void main (String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

Discussion



- Does Hello World have an architecture?
 - **Yes**
 - What does it look like?
 - **No**
 - Why not?

The Architecture of "Hello World"

```
public class HelloWorld {
```

```
    public static void main (String[] args){  
        System.out.println("Hello World");  
    }  
}
```

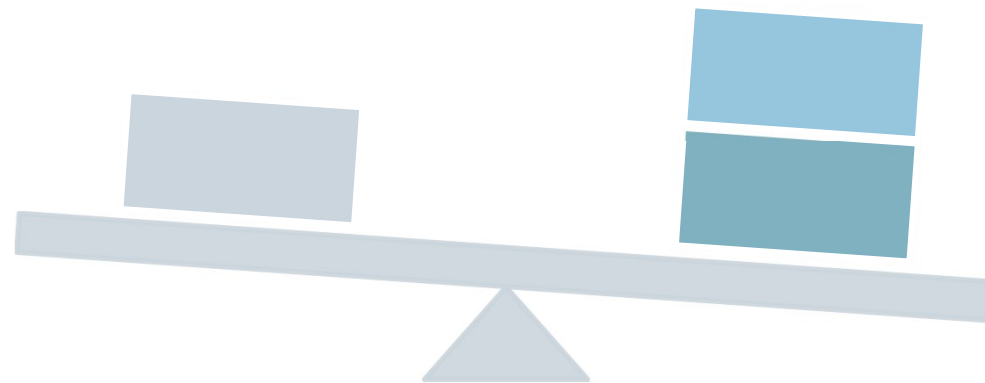
```
#include <iostream>  
using namespace std;
```

```
int main()  
{  
    cout << "Hello World";  
    return 0;  
}
```


Architecture Design Decisions

- **Design Decisions Balance** competing concerns
- **Some Design Decisions** are made **early** in the lifecycle
 - Typically have **far-reaching** effects
 - Are **hard to change** (in later phases or future projects)

→ The impact of architecture design decisions has to be known!

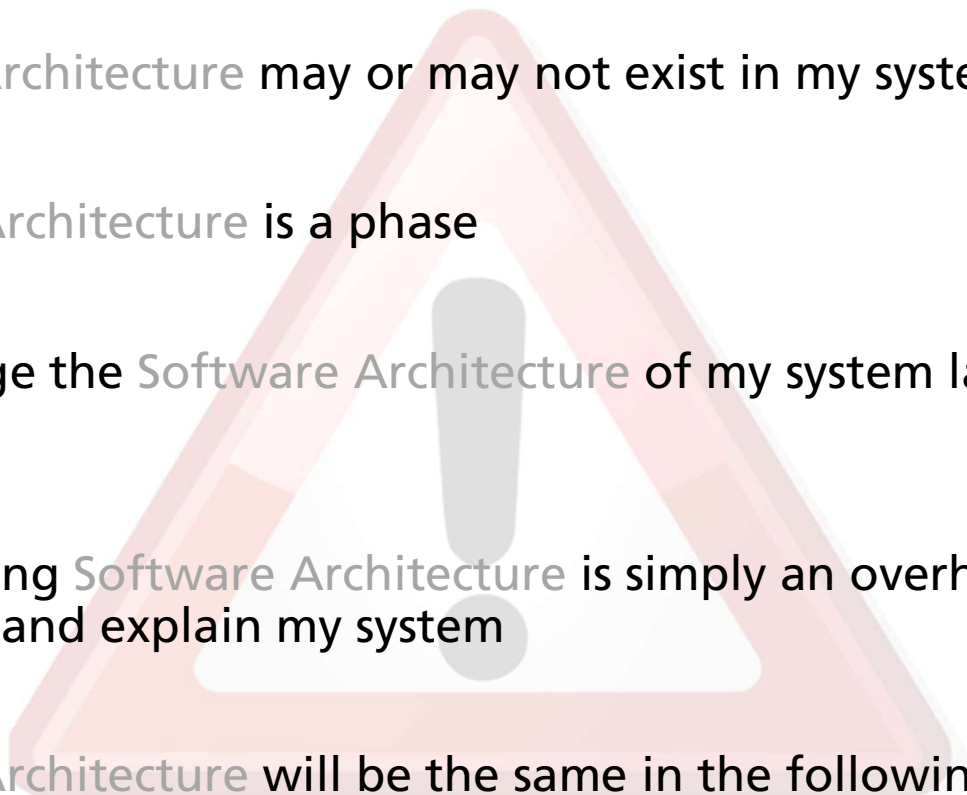


Examples of Design Decisions

- Programming language Java
- One central database
- No central instance of data, everything is distributed
- Three Tier Architecture
- Usage of an app generation framework for serving iOS and Android devices
- XML as data format
- Compression of data between client and server due to low network bandwidth
- Outsourcing of implementation of a component
- ...

Wrap Up

Common Misconceptions

- 
- Software Architecture may or may not exist in my system
 - Software Architecture is a phase
 - I can change the Software Architecture of my system later, whenever needed
 - Documenting Software Architecture is simply an overhead; I can always remember and explain my system
 - Software Architecture will be the same in the following projects
 - Software Architecture has nothing to do with my coding

Further reading: **Top 10 software architecture mistakes**

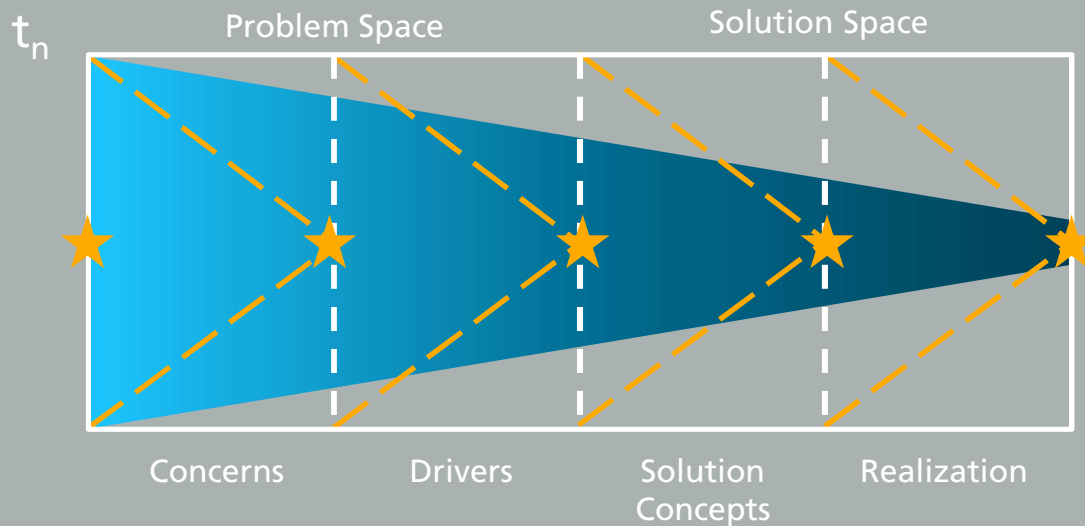
<http://www.infoq.com/news/2007/10/top-ten-architecture-mistakes>

Architecting in a Nutshell

Information Flow in Software Engineering



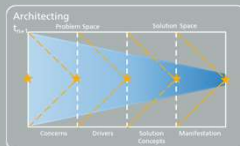
Architecting



Evolution

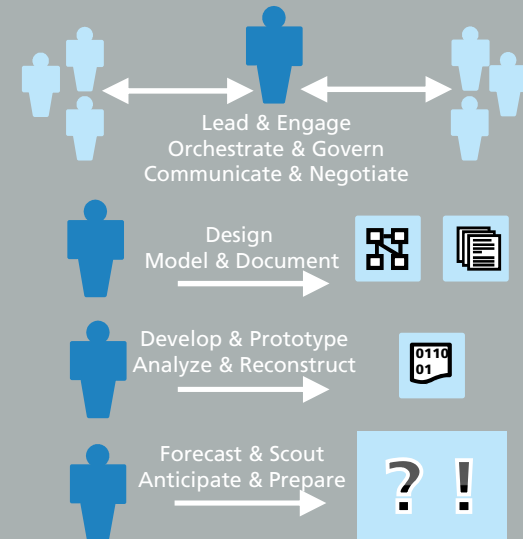


Change happens,
Systems follow!



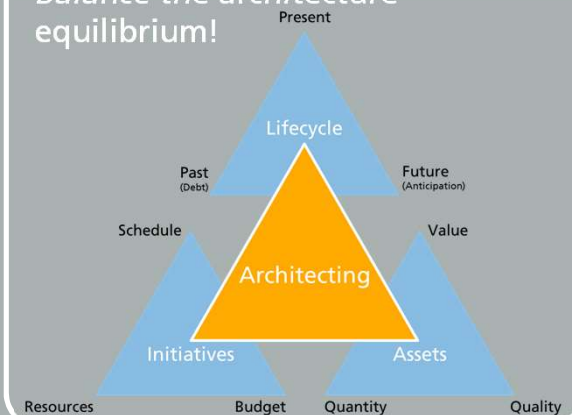
- Clear Technical Debt of the *Past*
- Prepare/Solve *Present* Challenges
- Anticipate *Future* Changes/Needs

Architect's Activities



Value Proposition of Architecting

Balance the architecture equilibrium!

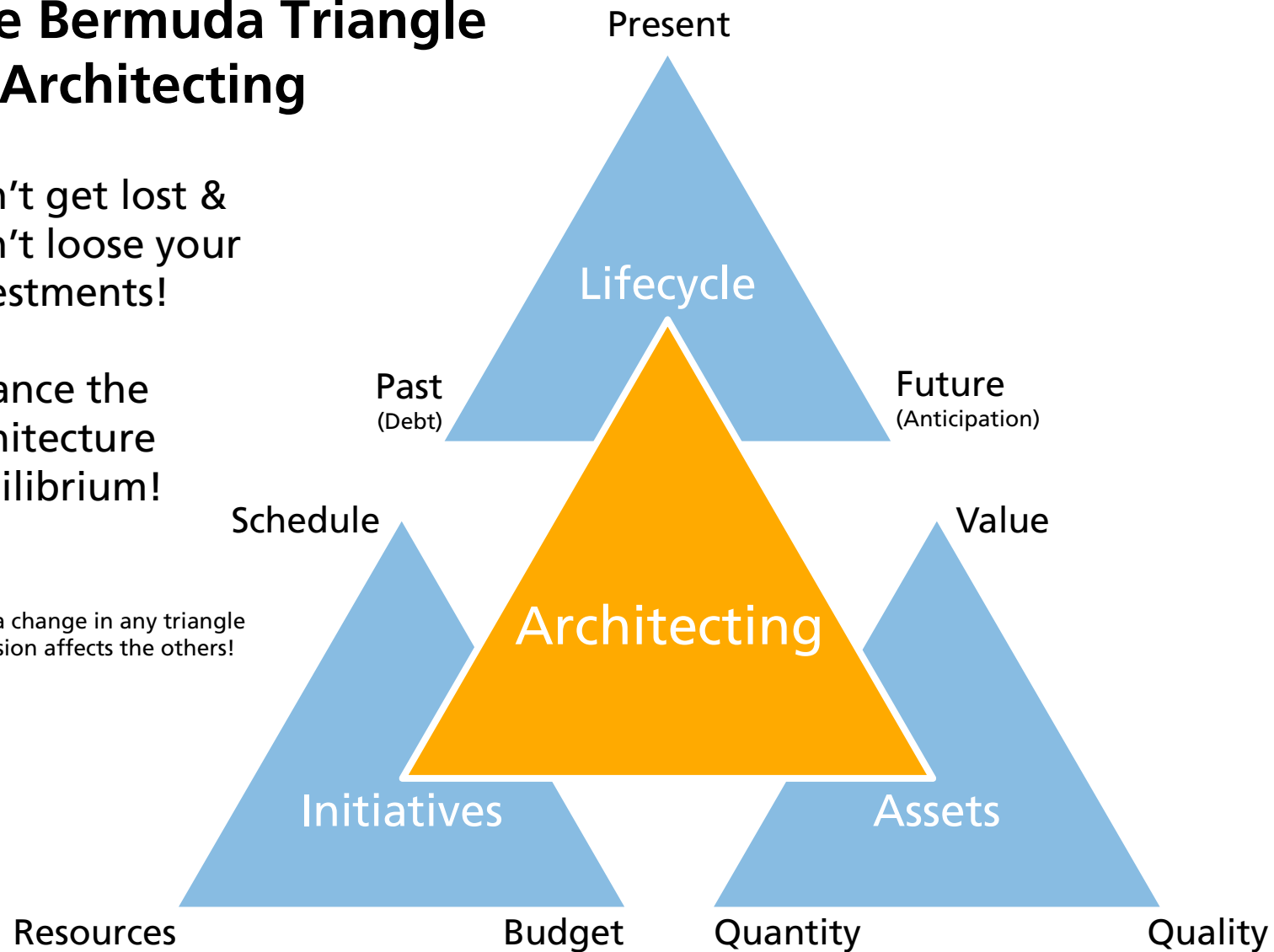


The Bermuda Triangle of Architecting

Don't get lost &
Don't loose your
investments!

Balance the
architecture
equilibrium!

Note: a change in any triangle
dimension affects the others!



Architectures...

■ ... **provide guidance**

- Plan for constructing a system
- Technical leadership and coordination
- Standards and consistency

■ ... **balance technical risks**

- Identification and mitigation
- Anticipation (preparation) for changes

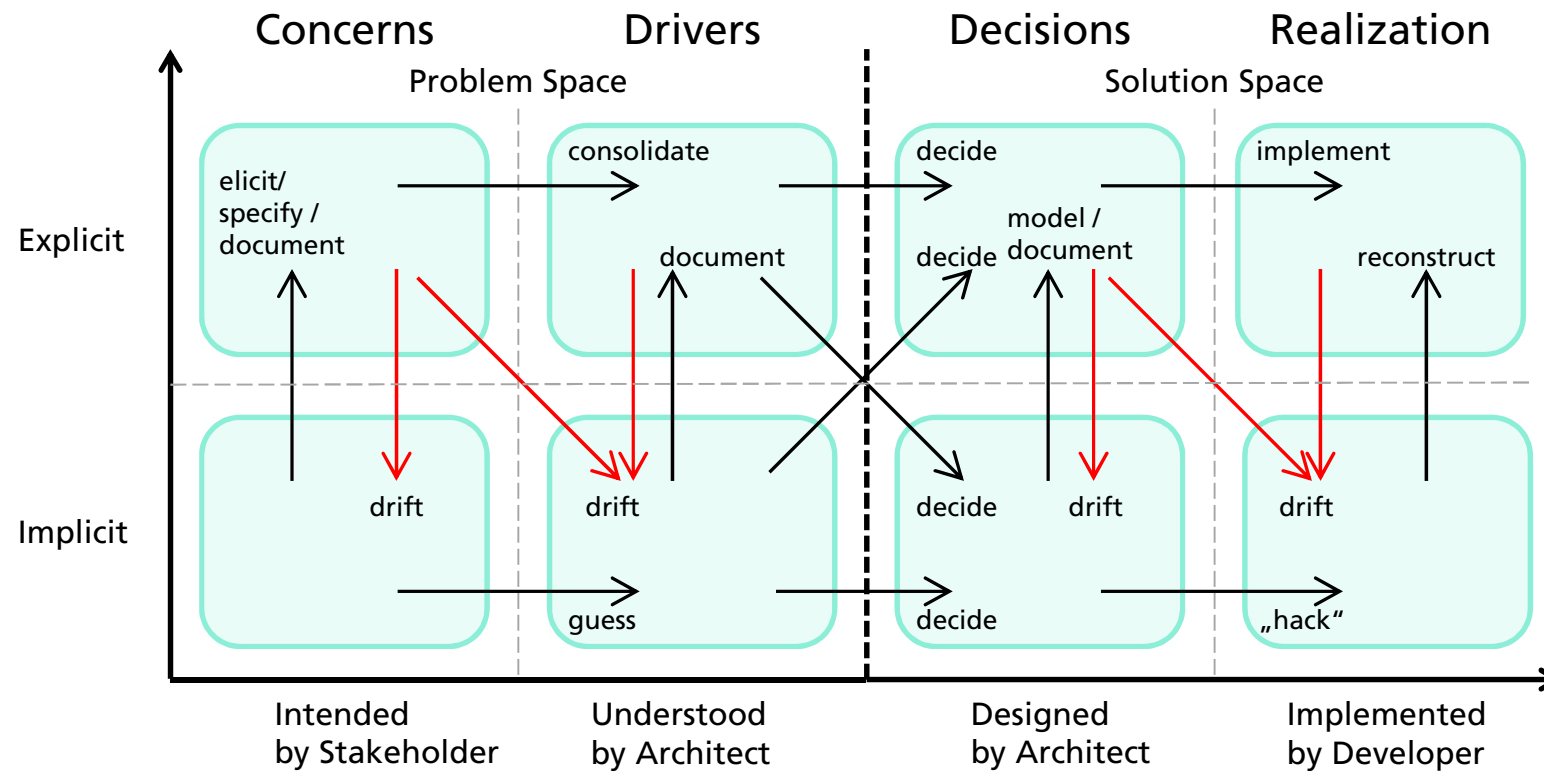
■ ... **enable communication**

- Clear technical vision and roadmap
- Explicit documentation for communication

■ ... **manage the inherent complexity** of software

- Products to be built
- Increasing interconnection of systems
- Integration with legacy systems
- Collaboration of organizational units

Evolution and Drift



Architecture Design Decisions

- **Design Decisions Balance** competing concerns
- **Some Design Decisions** are made **early** in the lifecycle
 - Typically have **far-reaching** effects
 - Are **hard to change** (in later phases or future projects)

→ The impact of architecture design decisions has to be known!

