



# Domain-Driven Design (DDD)

**Architectural Thinking for Intelligent Systems**

**Winter 2019/2020**

**Prof. Dr. habil. Jana Koehler**

# Agenda

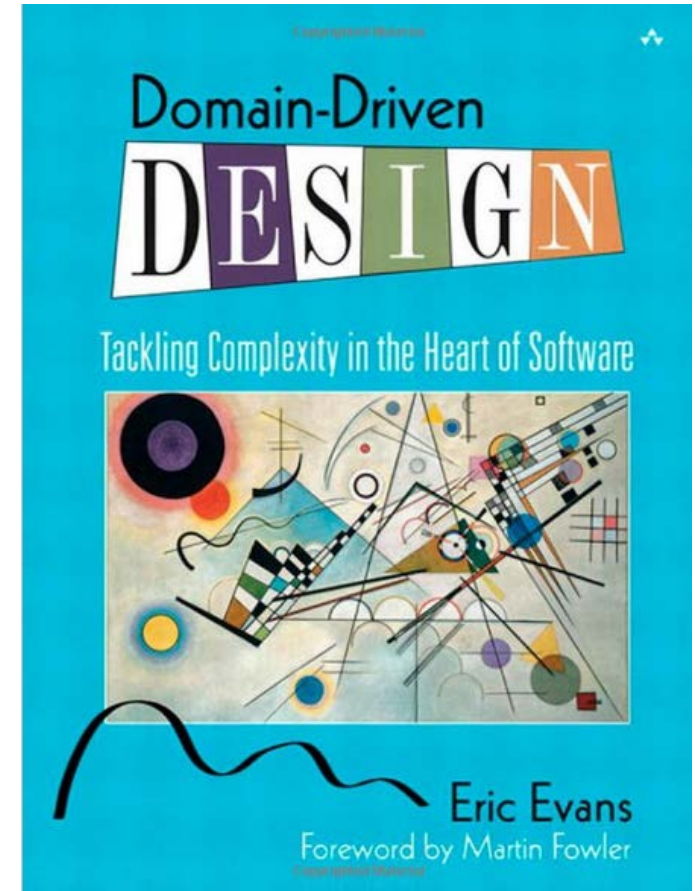
- Build an understanding the business domain
- Capture business understanding in models
- Separate business and application layer
- Domain elements and bounded contexts
  - Entities
  - Value objects
  - Domain events
  - Services
- DDD context maps and the big ball of Mud

## Tutorial Assignment 7:

- We analyze the business domain in more detail and identify the essential elements of the domain.
  
- We decide which
  - entities,
  - domain events, and
  - serviceswill be in the focus of our attention and begin to model these elements.

## From Business Case to System Structure

- Understand the business domain and reflect it in the architecture
- Speak the language of the business
- Find out about potential conflicts in the understanding of what needs to be done among different stakeholders
- Address risks & find compromises



## Understanding the Business Language

- Adopt the business language and use it in architecture, design, and code
  - Business languages are always spoken in a context
  - Clarify the meaning of specific terms in this context
- Create models to capture key concepts of the application domain at an adequate level of abstraction
- Code implements the model
  - «*express the model through the code*»
- Continuous Integration means above all ensuring a common understanding of the business language among the team!

## Example

- «Case"»
- «Study variables"»
- «Variable values"»

***"Tightly relating the code to an underlying model gives the code meaning and makes the model relevant."***

```

StudyValues
├── AlterHaupttaeterIn.cs
├── AlterKind.cs
├── AnzahlTaeter.cs
├── BezugHaupttaeterInZuKind.cs
├── FormDerGefaehrdung.cs
├── Geschlecht.cs
├── Haeufigkeit.cs
├── Language.cs
├── Lebenssituation.cs
├── Profession.cs
├── QuelleDerMeldung.cs
├── TrioWert.cs
├── Wohnkanton.cs
├── Zeitpunkt.cs
StudyVariables
├── AngabenZumBetroffenKind.cs
├── AngabenZurTaeterschaft.cs
├── Fall.cs
├── Fallbearbeiter.cs
├── InterventionLeistung.cs
├── Meldung.cs
├── Organisation.cs
├── PrimaereKindeswohlgefaehrung.cs
├── VermittlungUeberweisung.cs
├── WeitereFormenDerKindeswohlgefaehrung.cs

```

Angaben zur fallbearbeitenden Person						Angaben zur Meldung der Kindeswohlgefährdung			Angaben zur primären Kindeswohlgefährdung			Weitere Formen der Kindeswohlgefährdung			
Fall	Alter	Geschlecht	Profession	Anzahl der Jahre im Berufsfeld	Anzahl Jahre in Organisation	Quelle der Erstmeldung	Datum	frühere Meldung	Form der Kindeswohlgefährdung	Beginn	Häufigkeit	psychische Gewalt	körperliche Gewalt	Vernachlässigung	sexuelle Gewalt
1	16	weiblich	Soziale Arbeit	4	1	Gesundheitswesen	13.11.2016	nein	Münchhausen Stellvertreter:yi	in diesem Monat	mehrmals regelmäßig	nein	nein	nein	ja
2	15	weiblich	Pädagogik	3	2	Eltern(teil)	17.10.2016	ja	körperliche Gewalt	in diesem halben J	mehrmals (unspezifisch)	nein	nein	nein	ja
3	75	männlich	Pädagogik	4	3	Verwandte und soziales Um	19.09.2016	nein	sexuelle Gewalt	vor mehr als einem J	mehrmals unregelmässig	ja	nein	ja	nein
4	15	männlich	Medizin	6	4	Pflegeeltern, Heim, Kindert:	15.09.2016	nein	Gewalt zw. Bezugspersonen	vor mehr als einem J	mehrmals unregelmässig	ja	nein	ja	nein
5	16	weiblich	Pädagogik	10	5	Schule und schulische Diens	20.11.2016	nein	sexuelle Gewalt	in dieser Woche	mehrmals regelmäßig	ja	nein	nein	nein
6	55	weiblich	Pädagogik	35	6	KESB	25.11.2016	nein	sexuelle Gewalt	in dieser Woche	einmal	ja	nein	nein	nein
7	19	männlich	Medizin	3	1	Sozialbereich	30.11.2016	nein	Gewalt zw. Bezugspersonen	in diesem Monat	mehrmals unregelmässig	nein	nein	nein	nein
8	20	männlich	Polizei	20	2	Gesundheitswesen	05.10.2016	nein	Vernachlässigung	in diesem halben J	mehrmals regelmäßig	nein	ja	nein	nein
9	25	männlich	Feuerwehr	3	3	Strafverfolgung	13.10.2016	nein	psychische Gewalt	vor mehr als einem J	mehrmals (unspezifisch)	nein	nein	nein	nein
10	35	männlich	Pädagogik	5	4	anonyme Meldung	17.10.2016	ja	Gewalt zw. Bezugspersonen	vor mehr als einem J	einmal	nein	nein	ja	ja
11	45	weiblich	Psychiatrie	7	5	andere Organisation, Fachp	17.09.2016	nein	sexuelle Gewalt	in dieser Woche	einmal	nein	nein	ja	ja

## Same Language Everywhere ...

- The stronger the agreement on the meaning of business concepts, the more viable the system!

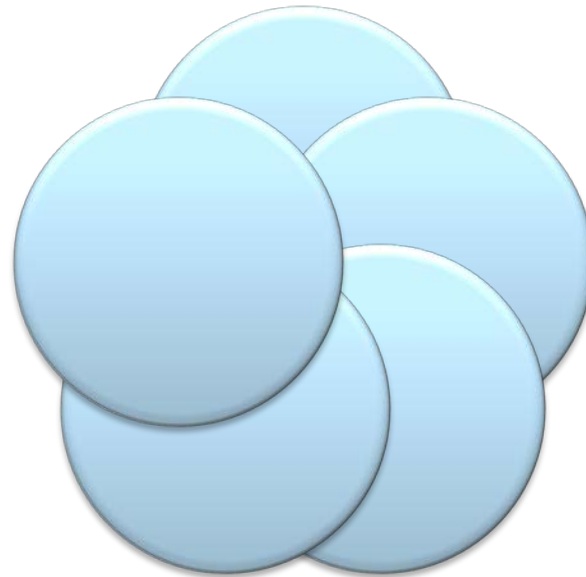
### (Business) Language

Architecture

Design

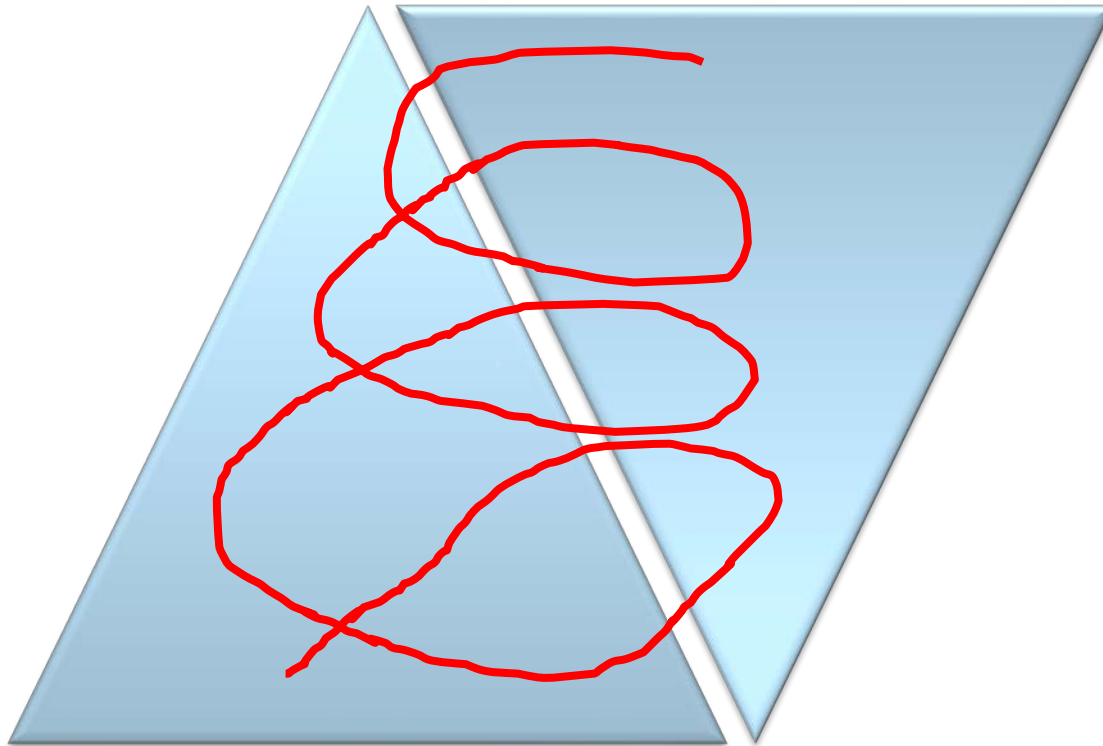
Code

Model



# Interleaving of Understanding – Modeling – Implementing

Documenting and understanding important concepts of the business domain



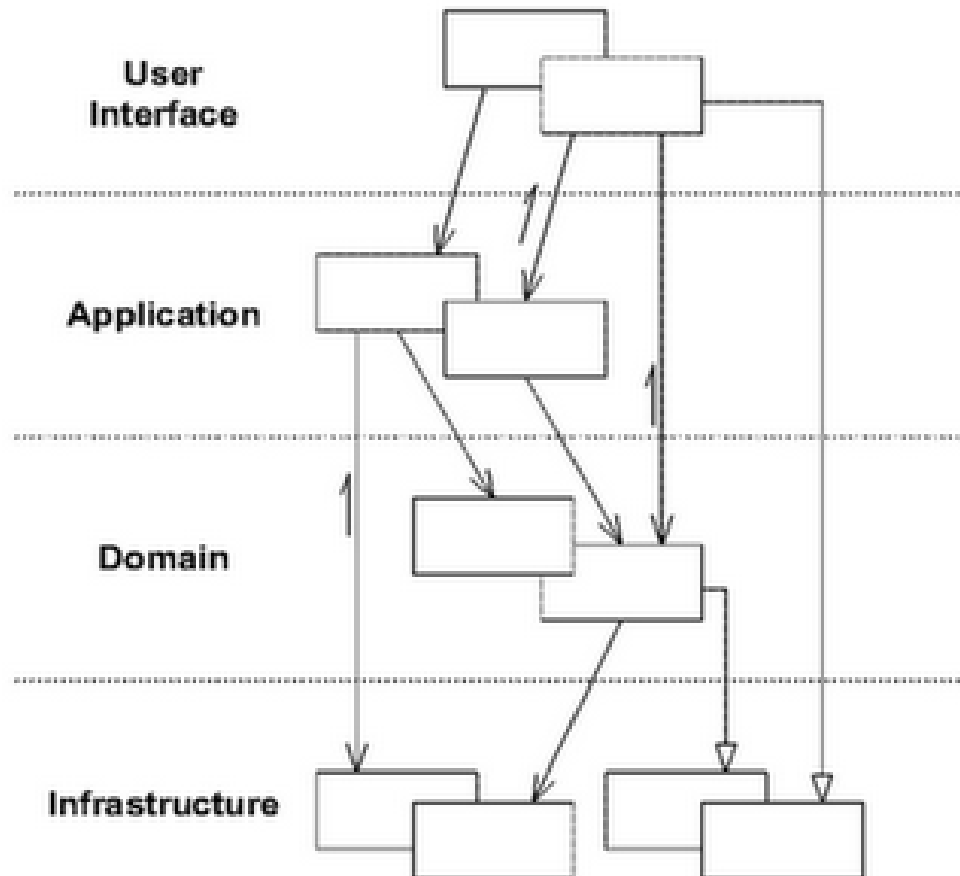
***"Sophisticated domain models seldom turn out useful except when developed through an iterative process of refactoring, including close involvement of the domain experts with developers interested in learning about the domain."***

Designing a part of the software to reflect the understanding of the business domain



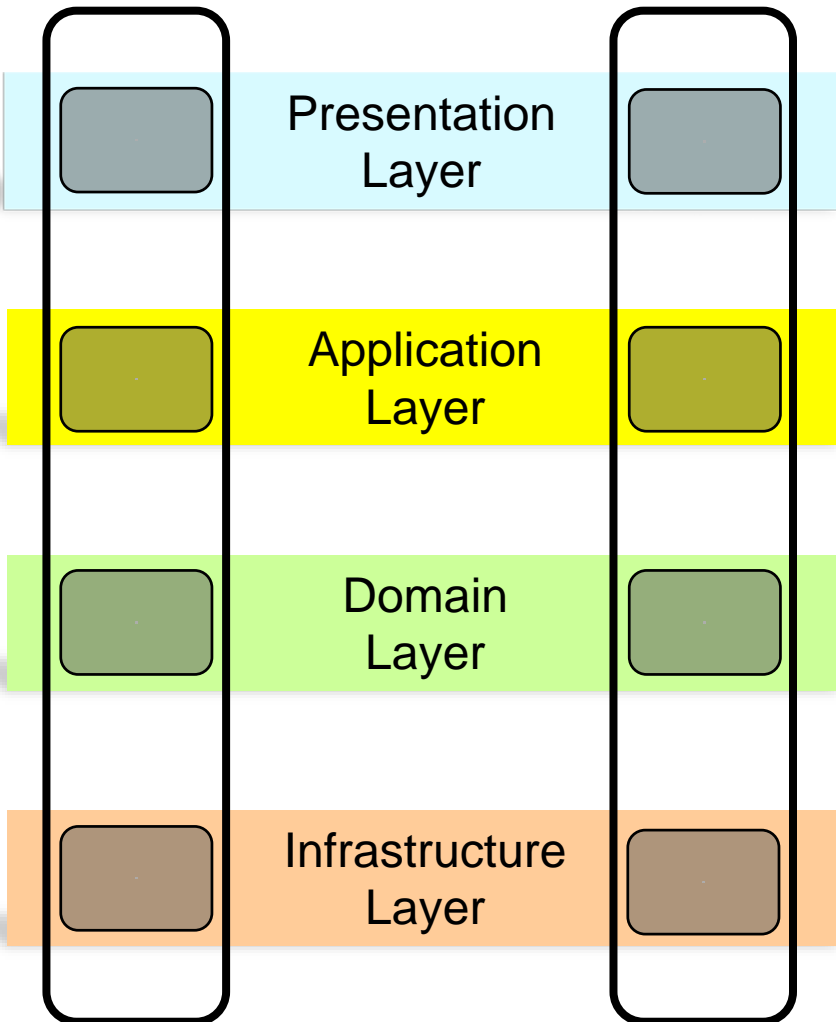
# Essential Elements of any Application Domain

- Entities
- Value Objects
- Domain Events
- Services



**Domain and application exist independently of IT TECHNOLOGY!**

## Divide and Conquer – Build Layers and Components



- Each layer provides clearly defined interfaces and uses services from underlying layers
- Each component implements a specific business or technical function
- **Understanding the business domain is crucial for architecture viability!**

## How to Proceed?

- Design the domain
- Do not establish dependencies on
  - Infrastructure
  - User interfaces
  - Application logic
- Split complex domains into subdomains
  - «Bounded Contexts»
- Domain objects have no application logic, but only logic that emerges directly from the business domain

### ➤ ISOLATION
















# Entities

- Things, people who have an identity = business objects
- Clearly defined (longer) life cycle
  - States relevant in the business domain
    - remember BPMN events
  - Rules control state transition
- Attributes may vary
  - Need for discussion and clarification
- One of the entities plays a particularly important role
  - Examples
    - Restaurant: bill
    - Bank: account
    - Social work (case management): case

# Value Objects

- Things that describe the state of entities
- No own identity, never composed of entities, no life cycle that would be meaningful and necessary from a business point of view

```
public class PrimaereKindeswohlgefaehrung
{
    0 references | Fux Etienne tcfux, 147 days ago | 1 author, 1 change
    public FormDerGefaehrdung FormDerGefaehrdung { get; set; }
    0 references | Fux Etienne tcfux, 147 days ago | 1 author, 1 change
    public Zeitpunkt Beginn { get; set; }
    0 references | Fux Etienne tcfux, 147 days ago | 1 author, 1 change
    public Haeufigkeit Haeufigkeit { get; set; }
}
```

- ▲  StudyValues
  - ▷  C# AlterHaupttaeterIn.cs
  - ▷  C# AlterKind.cs
  - ▷  C# AnzahlTaeter.cs
  - ▷  C# BezugHaupttaeterInZuKind.cs
  - ▷  C# FormDerGefaehrdung.cs
  - ▷  C# Geschlecht.cs
  - ▷  C# Haeufigkeit.cs
  - ▷  C# Language.cs
  - ▷  C# Lebenssituation.cs
  - ▷  C# Profession.cs
  - ▷  C# QuelleDerMeldung.cs
  - ▷  C# TrioWert.cs
  - ▷  C# Wohnkanton.cs
  - ▷  C# Zeitpunkt.cs

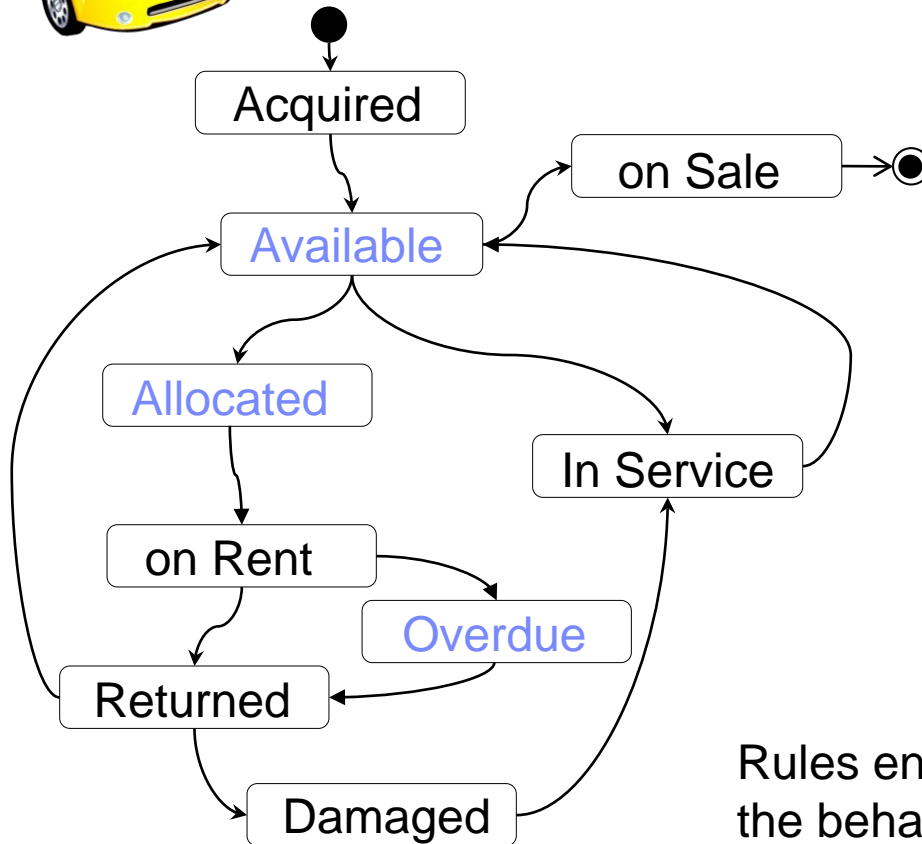
## Domain Events

- Which events of the domain determine the life cycle of entities?
- Which events cause state changes?
  - who/what triggers the event?
  - when does the trigger occur?
  - which properties characterize the event?
- Important: Separate business events from application-specific (architecture-specific) events
  - a case of child abuse is suspected
  - a case of child abuse has been edited in the data base by an agency

## Services

- Procedures and processes, which change the state (values of properties) of entities
- Understanding business processes: business services/business capabilities
- Service contract
  - Who can use the service when and how?
  - What does the service guarantee?
- Examples:
  - Restaurant ...

# Derive Services from the Lifecycle of Entities



*“A car from another branch may be allocated, if there is a suitable car available and there is time to transfer it to the pick-up branch.”*

*“If a car is three days overdue and the customer has not arranged an extension, insurance cover lapses and the police must be informed.”*

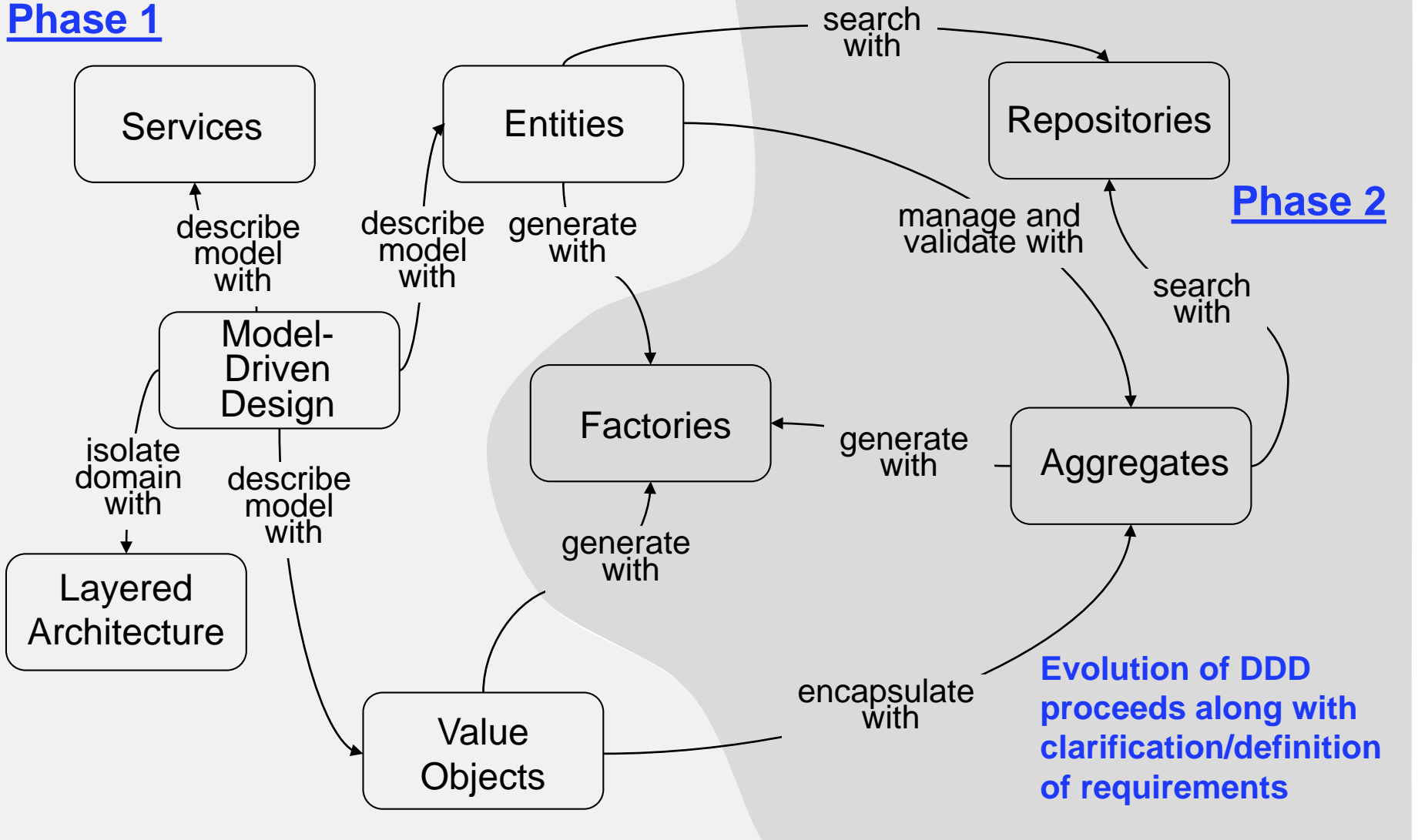
Rules ensure the integrity of data and regulate the behavior of actors

➤ and are a source for acceptance tests!!!



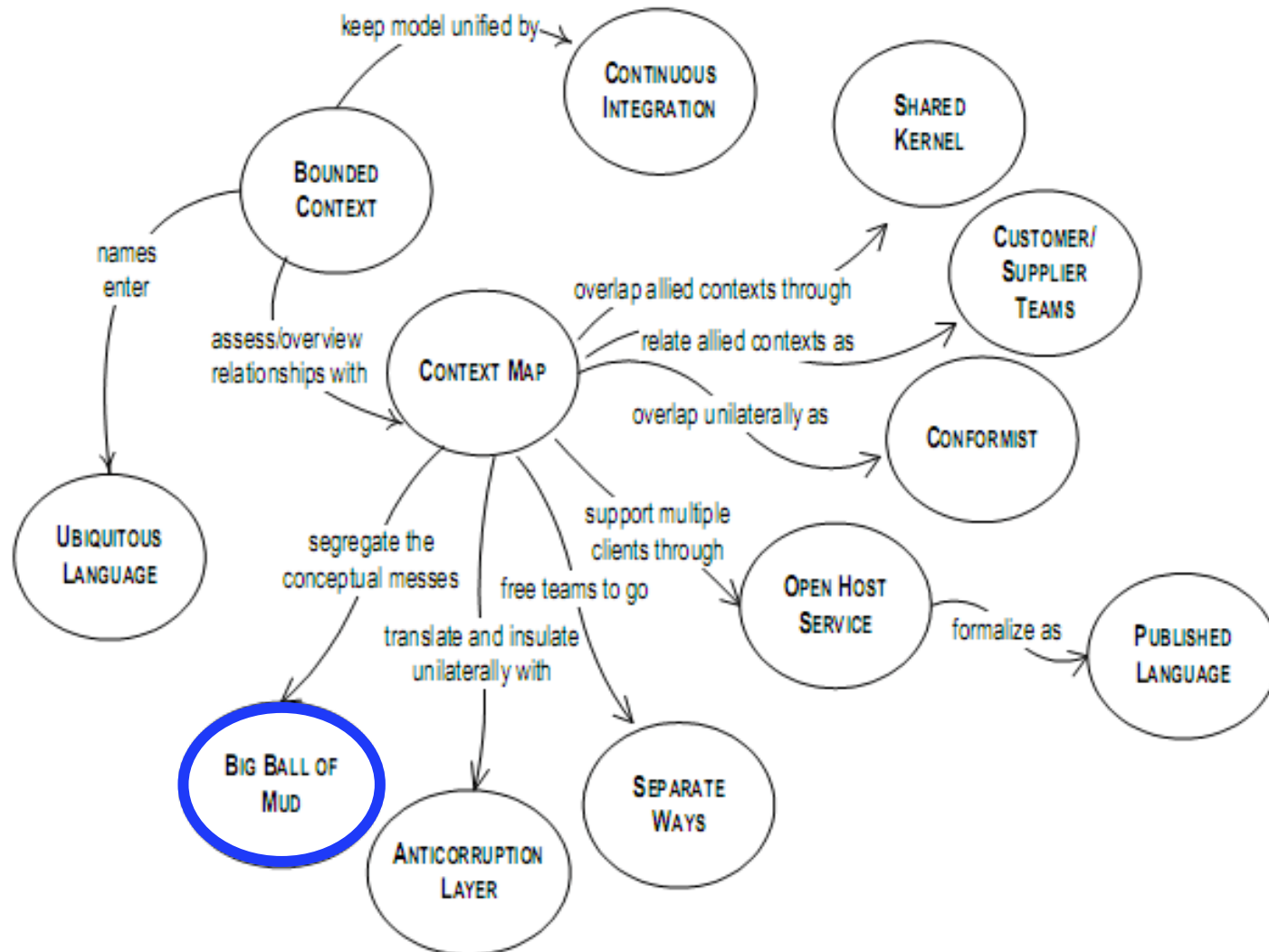
# Further Elements of Domain-Driven Design

## Phase 1

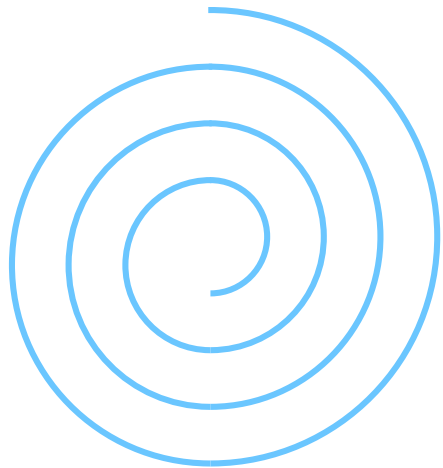


# Using DDD in Large Project

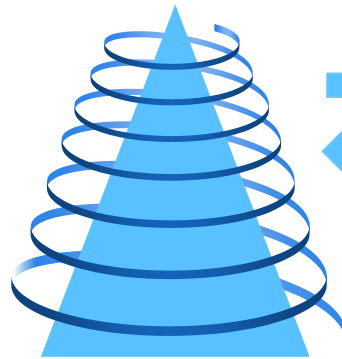
## Maintaining Model Integrity



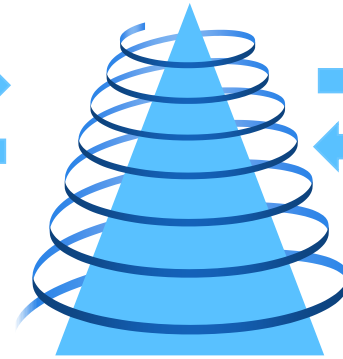
# Where are we in Architectural Thinking?



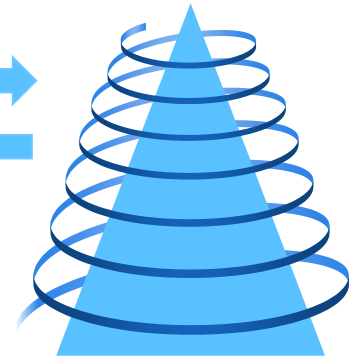
Goal hierarchies,  
Scenarios, system vision  
Context view



Requirements  
Constraints

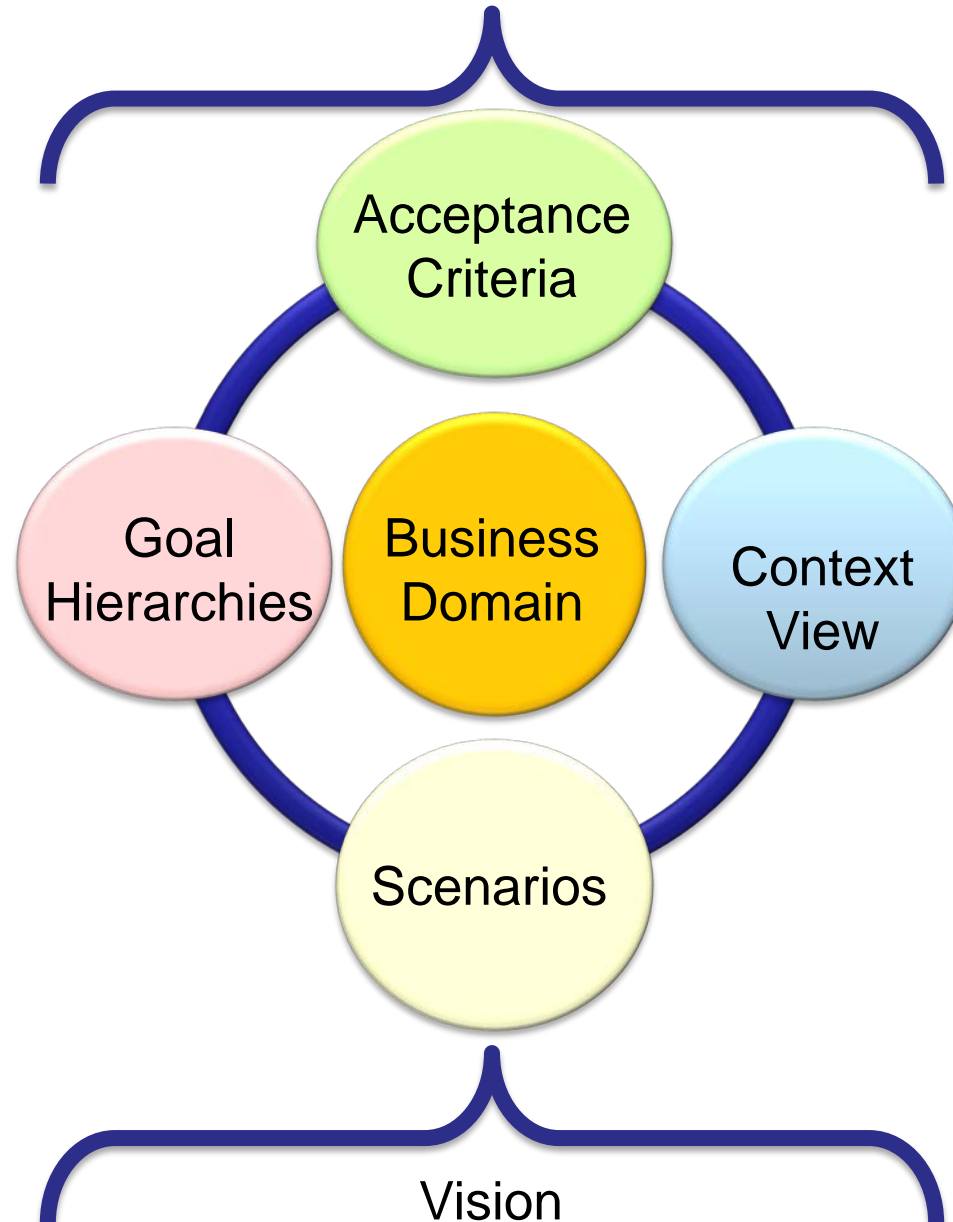


Architectural Draft  
System Idea

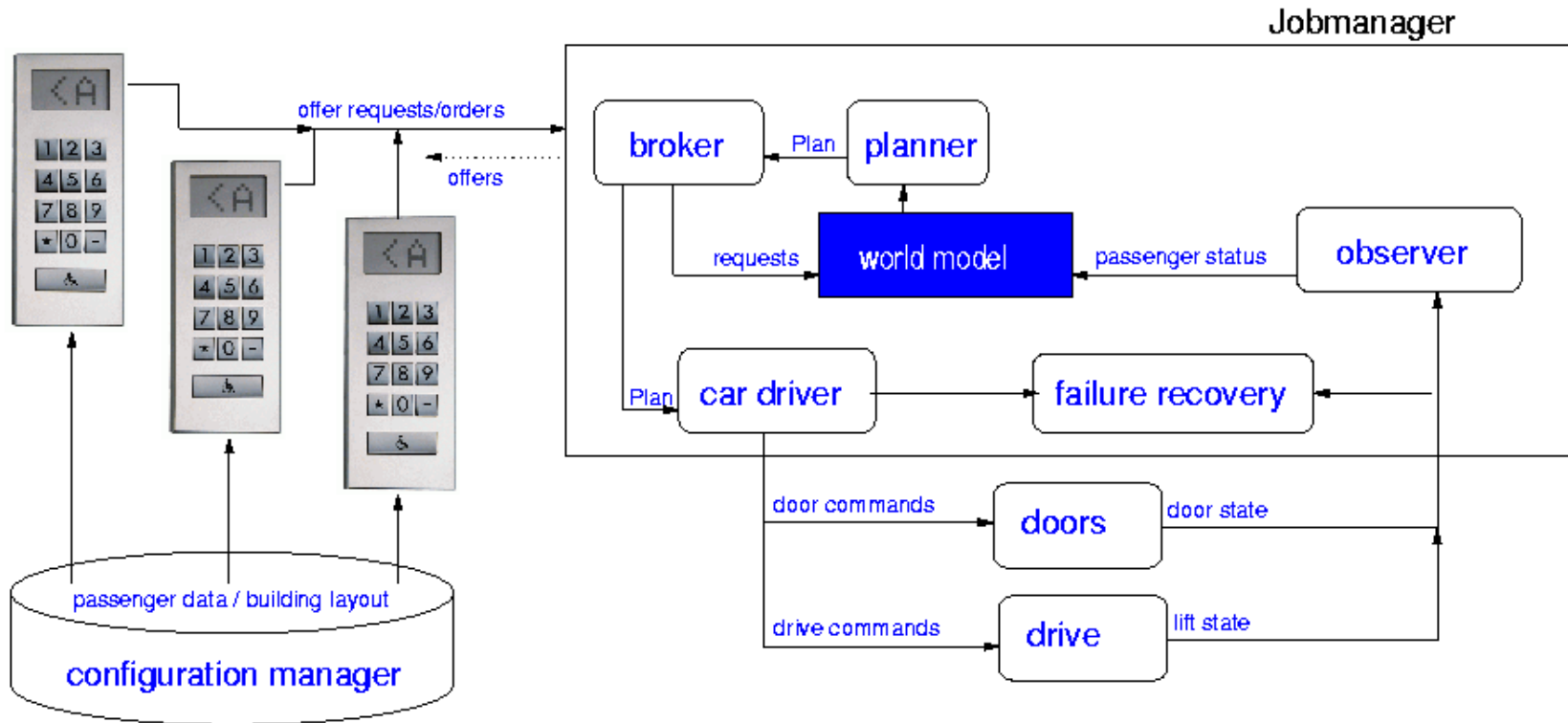


Design

Models of the Business Domain

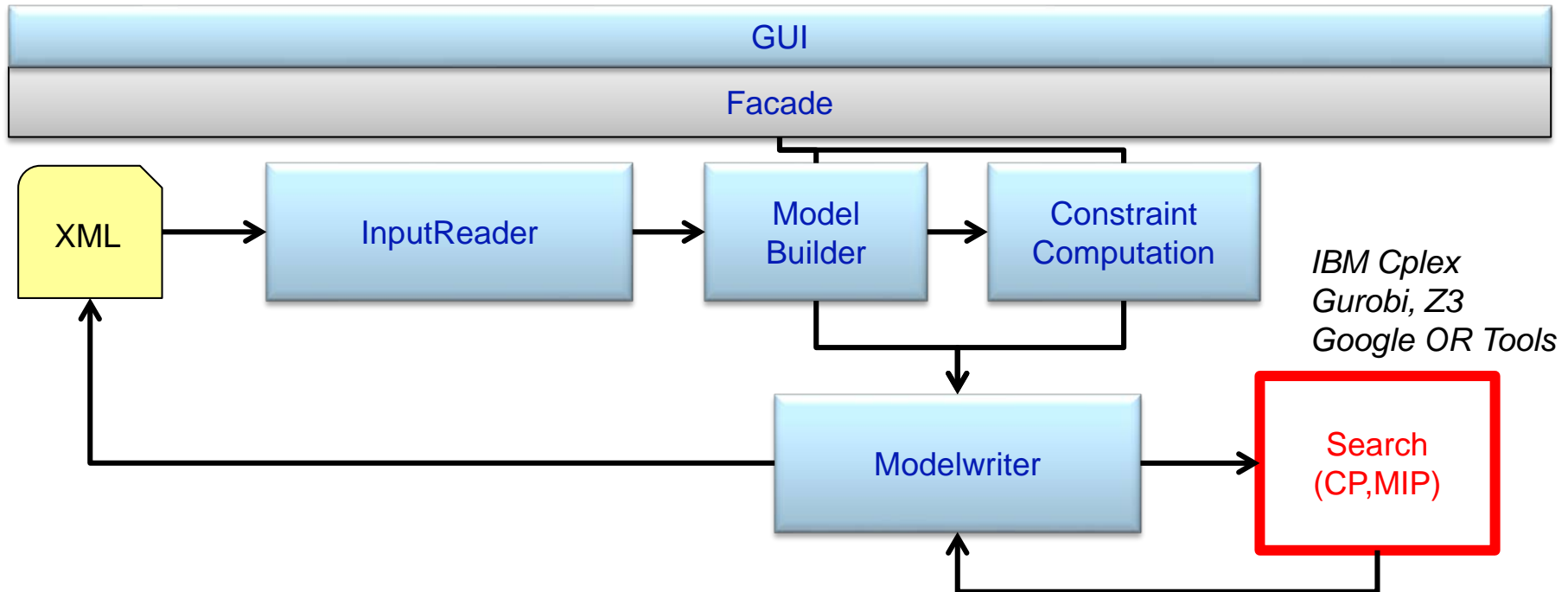


# DDD for Elevator Destination Control

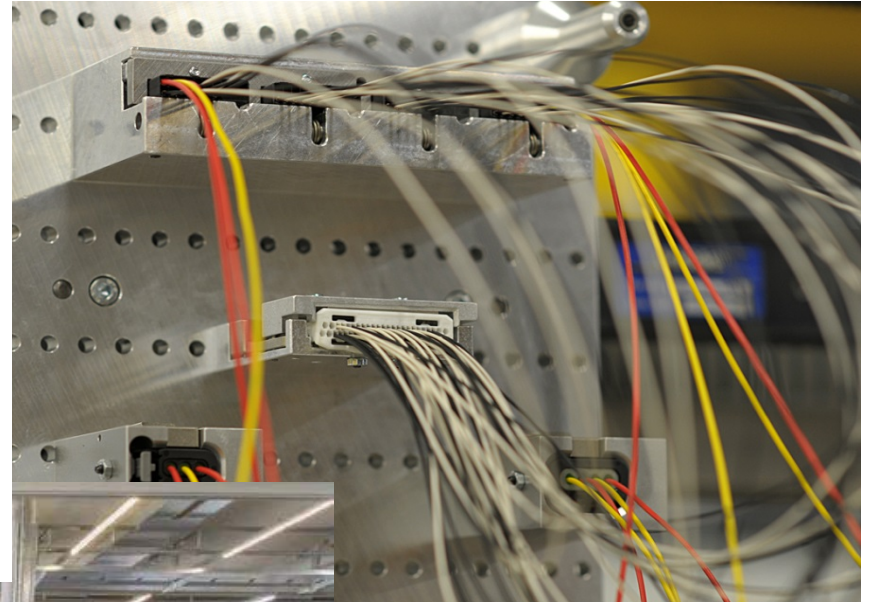
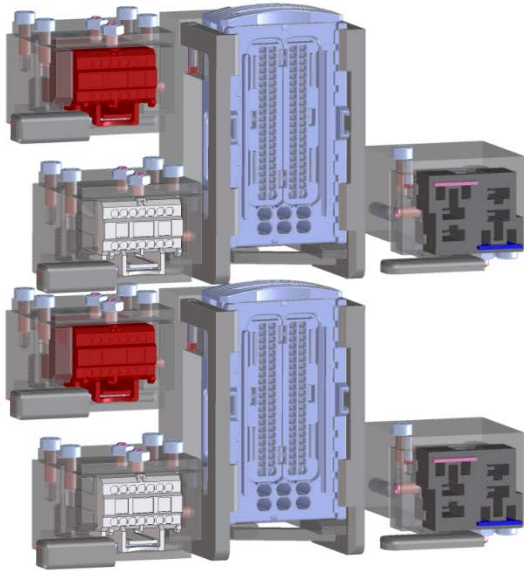


# AI System for Cable Wiring

- 10 Code Contributors
- Lines of C# Code: 4500 core system, 5300 GUI, 1800 tests
- > 1200 commits 2012-2016, test coverage 65%
- Yearly major refactoring, maintainability index > 85%



# Understanding a Business Domain



# Summary

- Adequate understanding of the business domain is crucial for architectural thinking
- Capture business understanding in models – create views for communication
- Separate business domain from application domain
- Clarification of business terms is critical and time-consuming to build a uniform understanding among stakeholders
- Undetected ambiguities or different semantic understandings are a source of major risks
- DDD focuses on business objects (entities) and their life cycle
  - clear separation of entities - value objects - events - services
- Business model should never be influenced by the infrastructure or technology used
- Large domains are broken into bounded contexts and linked via a context map



## Working Questions

1. What do we understand by Domain-Driven Design (DDD)?
2. What is the most important goal of DDD?
3. Why should one strictly separate Domain Layer and Application Layer?
4. What role does Continuous Integration play in DDD?
5. What is a context in DDD?
6. Give examples of entities, value objects, domain events, and services in a domain of your choice.
7. What role does the context map play and how do contexts relate to the Big Ball of Mud?
8. Why do we discuss DDD although our topic is software architecture?