Translation of Use Case Scenarios to Java Code

Michał Śmiałek, Norbert Jarzębowski, Wiktor Nowakowski

Warsaw University of Technology

Krajowa Konferencja Inżynierii Oprogramowania

Kraków, Poland, September 10-13, 2012

Presentation outline

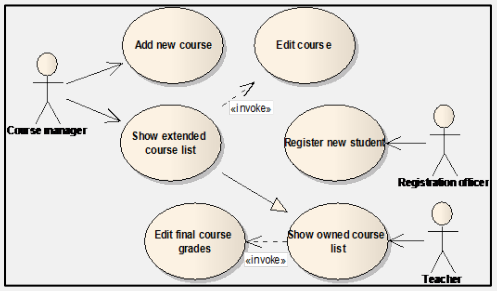
- Introduction
- Short introduction to RSL (use case scenario language)
- Side note: complexity of software systems
- Transformation rules
- Transformation program
- Validation and conclusion

The problem

- Use cases are quite imprecise
- Not possible to automate the transition from requirements to code
- Quite sparse attempts to treat use cases as “first class citizens” in software development
- Idea: bring use cases closer to code

Solution overview

Use Case model

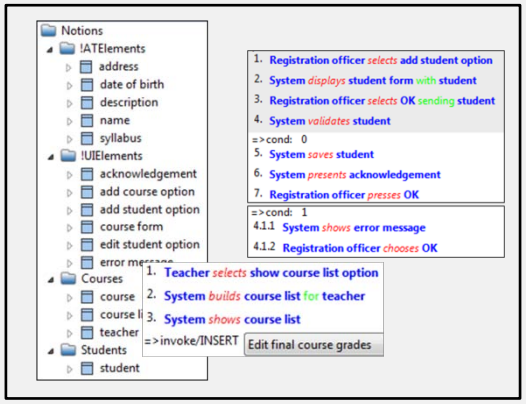


Transformation Rules

Java code

MOLA program

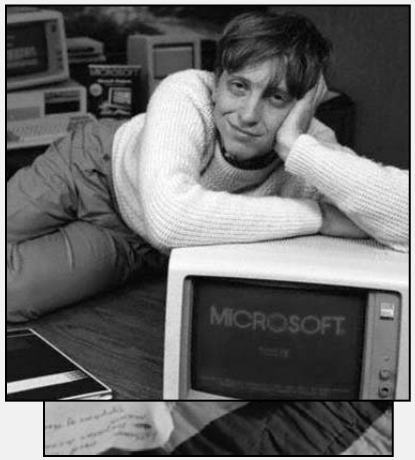
RSL scenario model

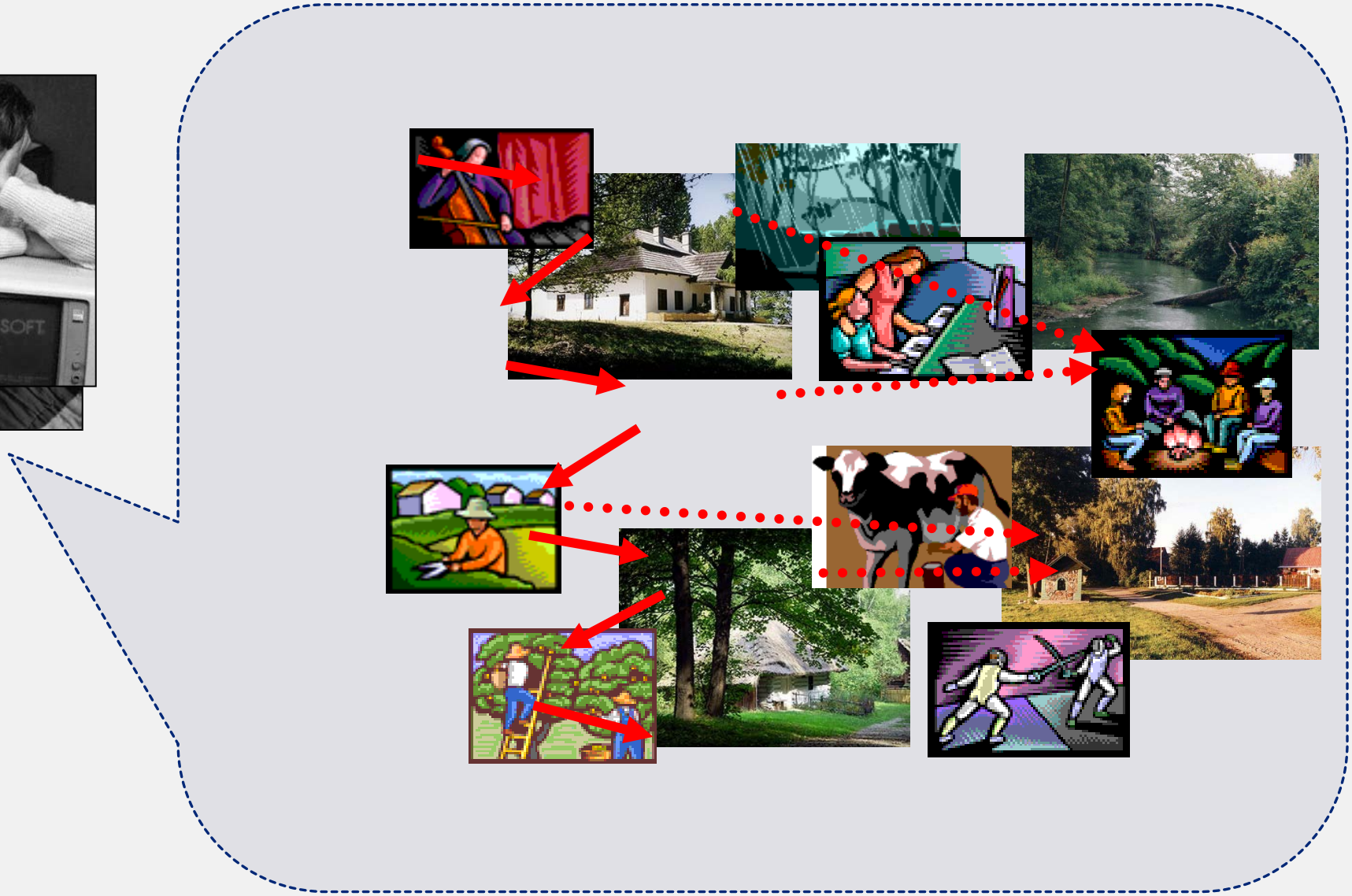


UML design model
+ Java method bodies

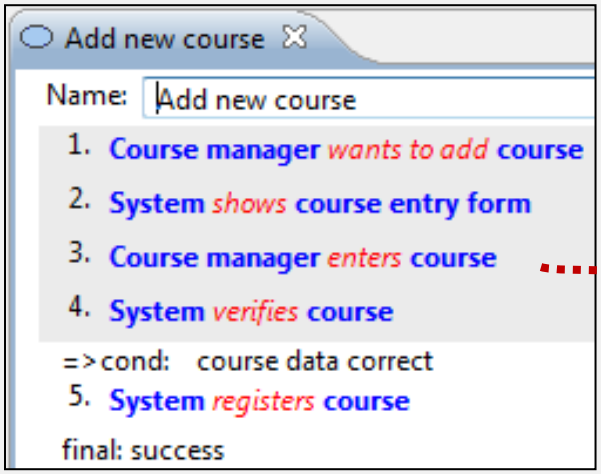
Short introduction to RSL

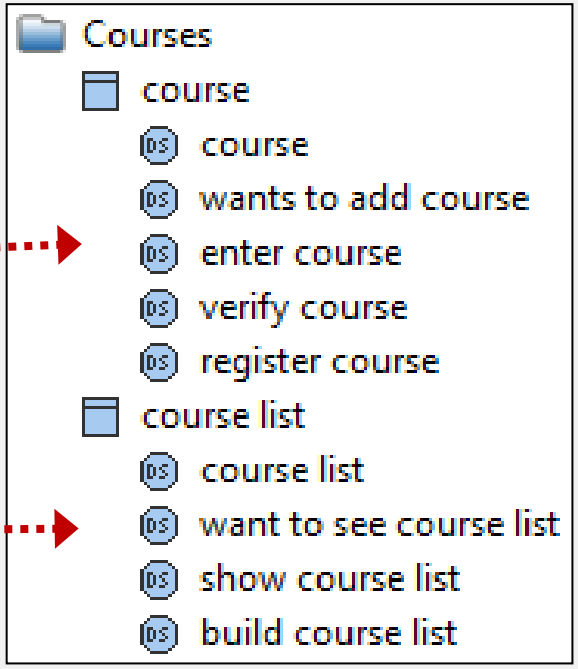
General idea: separate the story from the domain

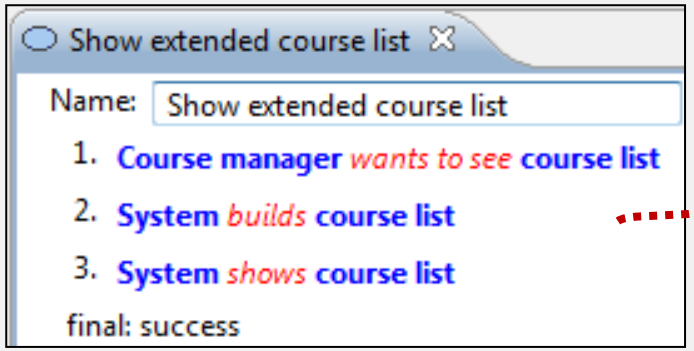




RSL = use case scenarios + domain vocabulary



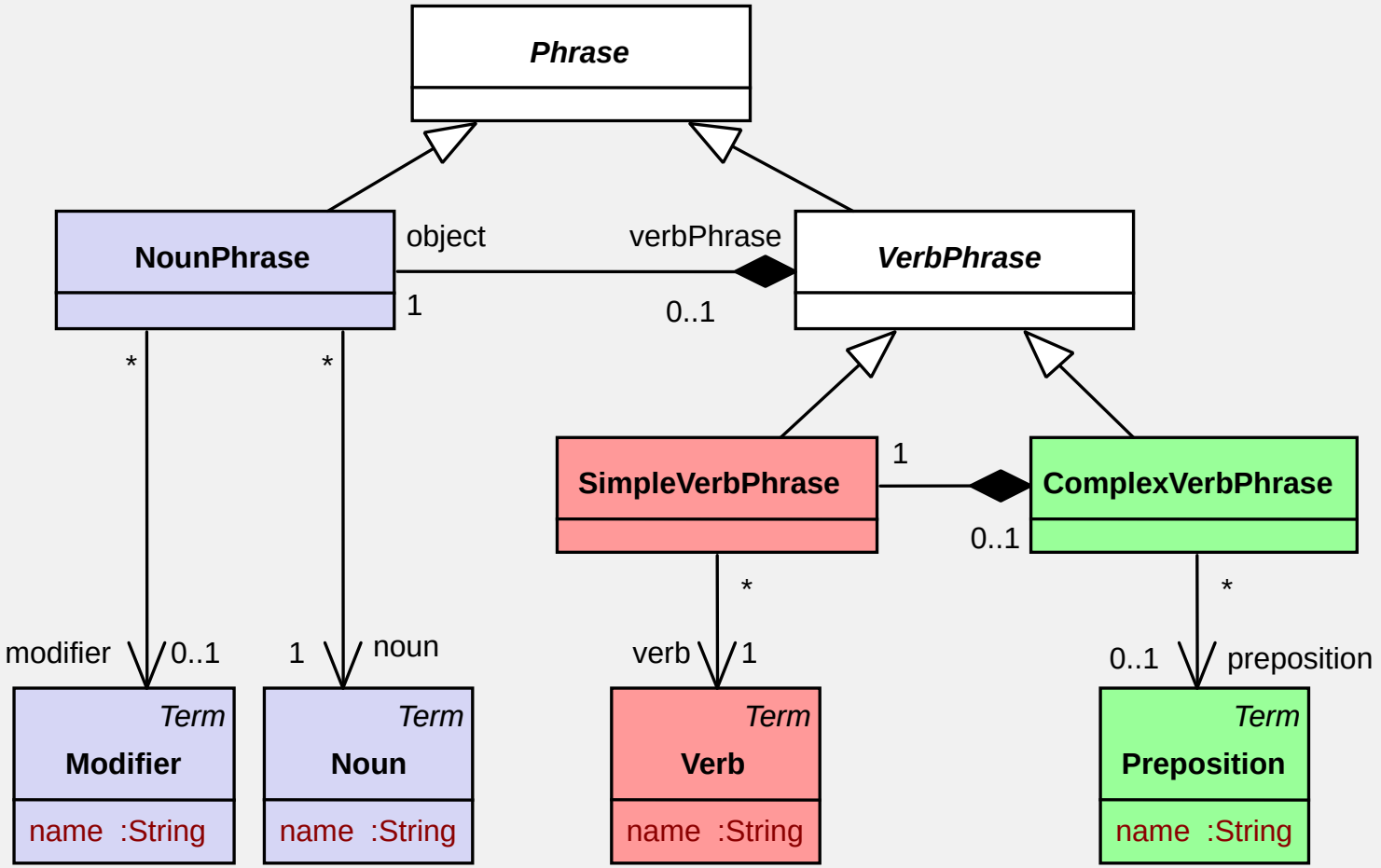


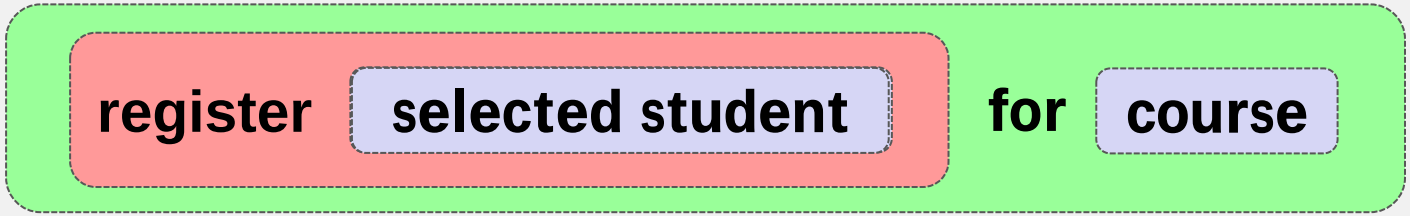


Scenarios refer to notions and phrases from the domain vocabulary

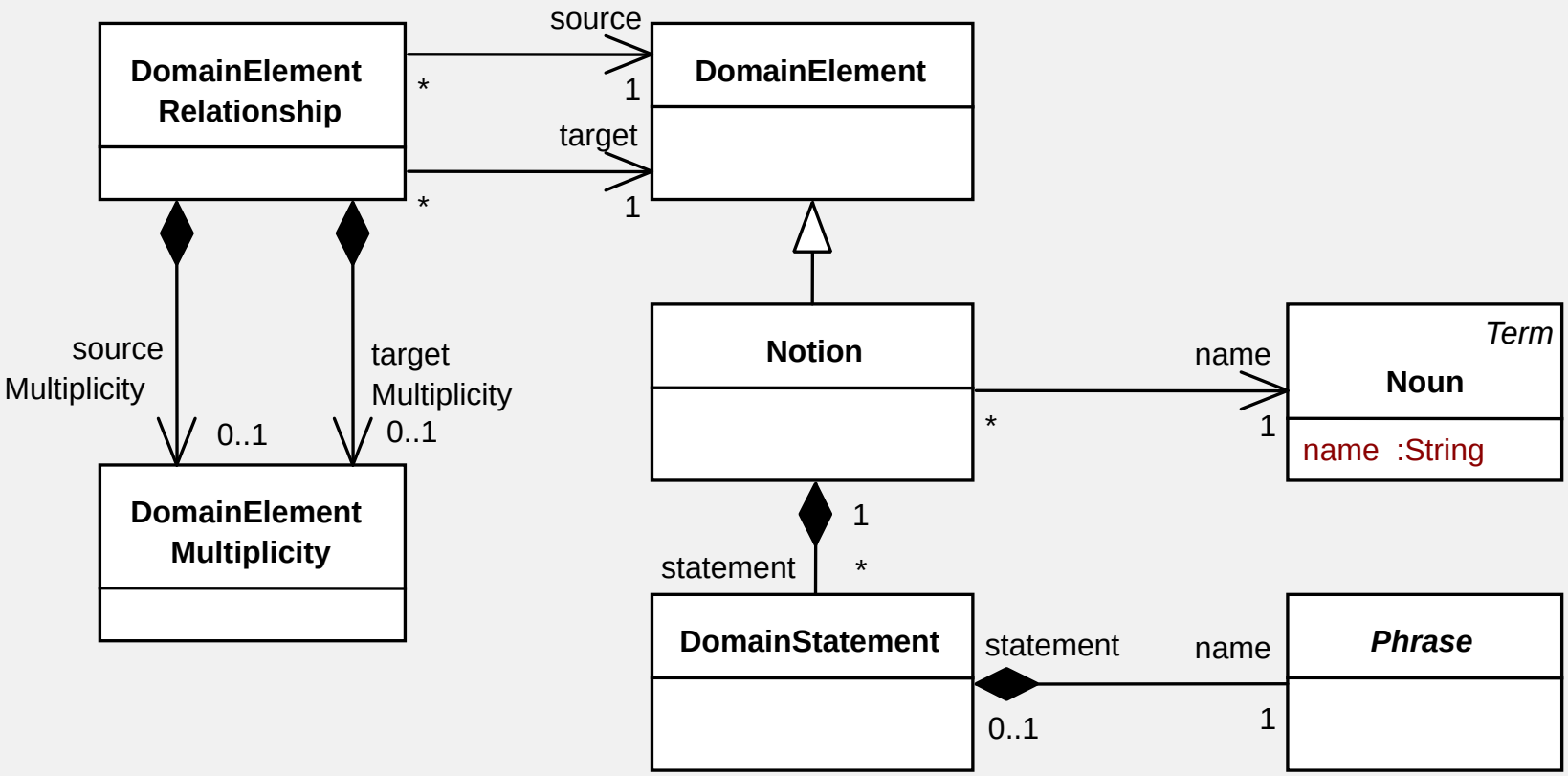
Domain vocabulary contains definitions of notions and phrases in the context of the system's domain

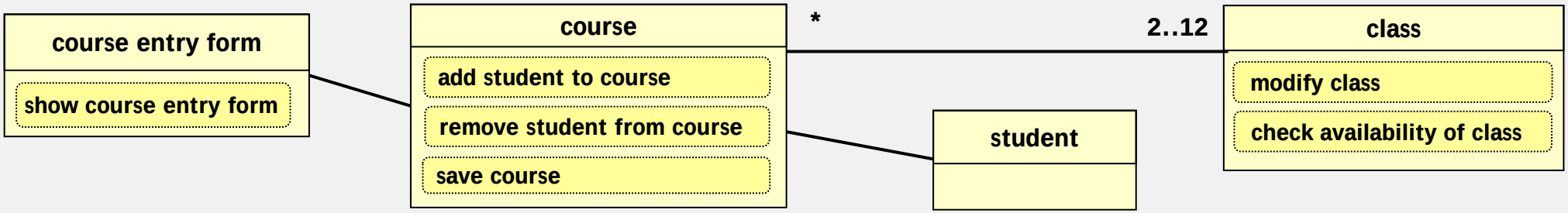
Phrases: basic building blocks in RSL





Grouping phrases within notions





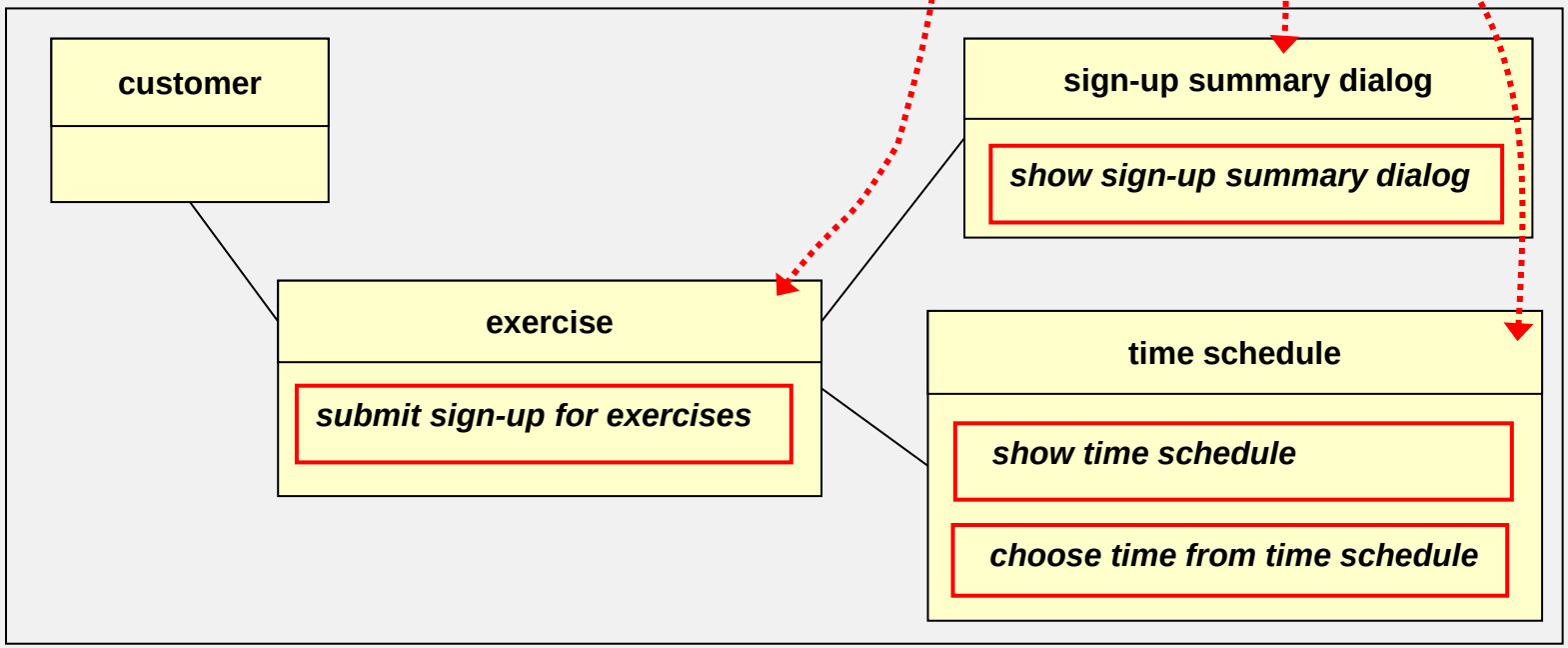
Full RSL model - example

Requirements representations

Basic scenario

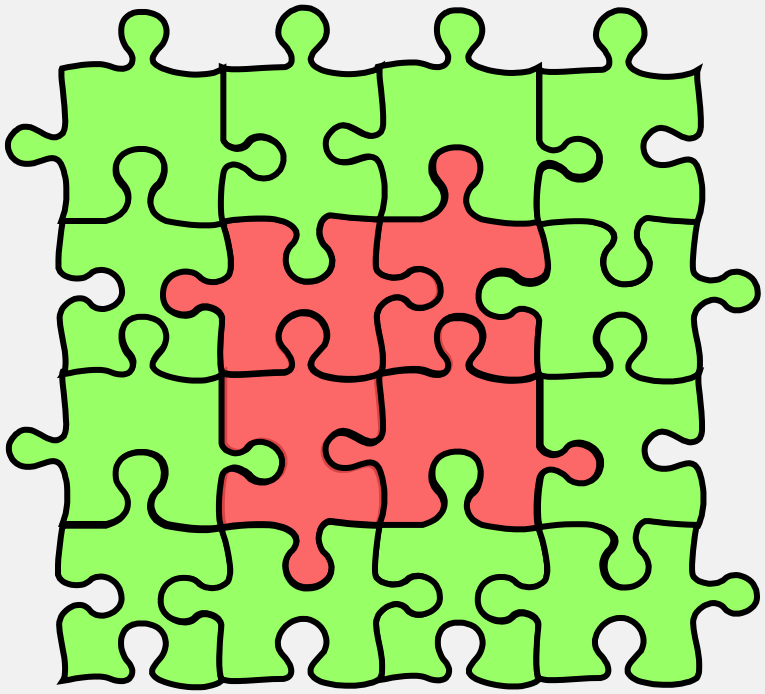
1. Customer *wants to sign up for* **exercises**.
2. System *checks* availability of **exercises**.
3. System *shows* **time schedule**.
4. Customer *chooses* time from **time schedule**.
5. System *shows* **sign-up summary dialog**.
6. Customer *submits* sign-up for **exercises**.
7. System *signs up* customer for **exercises**.

Domain vocabulary



Side note: complexity of software systems

Essential and accidental complexity



„The ***essence*** of a software entity is a construct of interlocking concepts: data sets, relationships among data items, algorithms, and invocations of functions.
[..] such a conceptual construct is the same under many different representations.” *

„[..] ***accidental*** tasks arise in representing the construct in language.” *

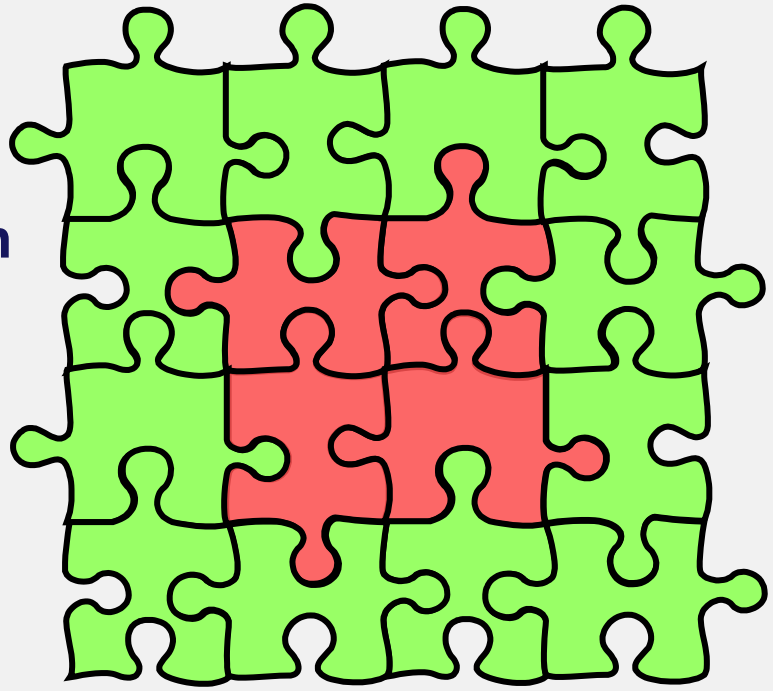
* F.P. Brooks, ”No silver bullet: Essence and accidents of software engineering”

The „essence” in contemporary systems

user interface

application logic

domain logic



In most well-designed multi-tier systems, the software essence is realized by the application logic and the domain logic.

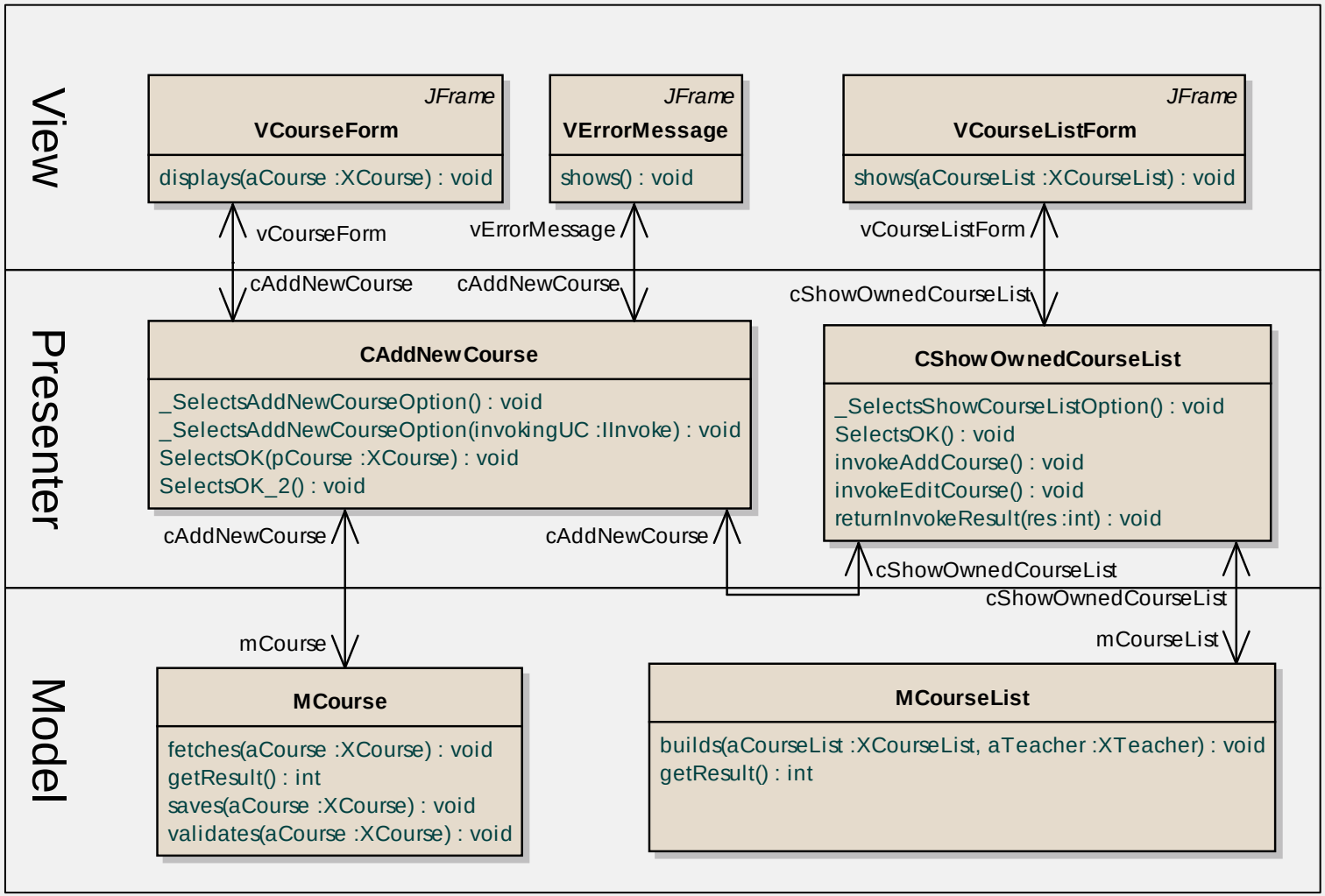
In typical development process, the essence of the system to be build is specified in details within a software requirements specification.

Transformation rules

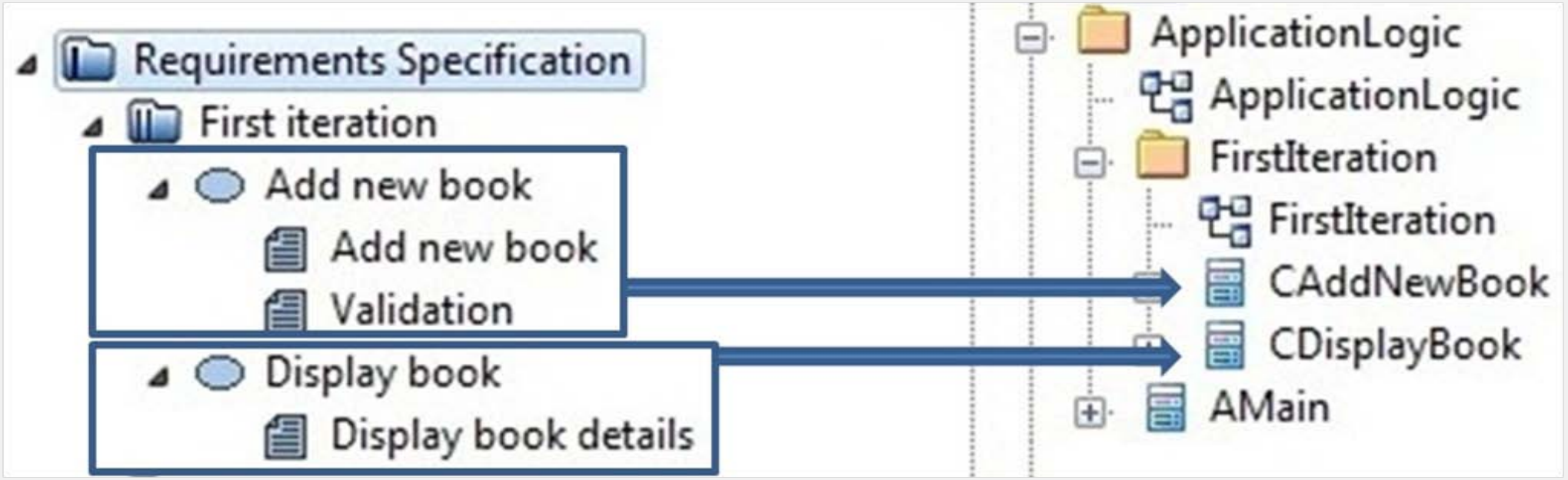
Overview of the approach: target architecture

application logic ⇢

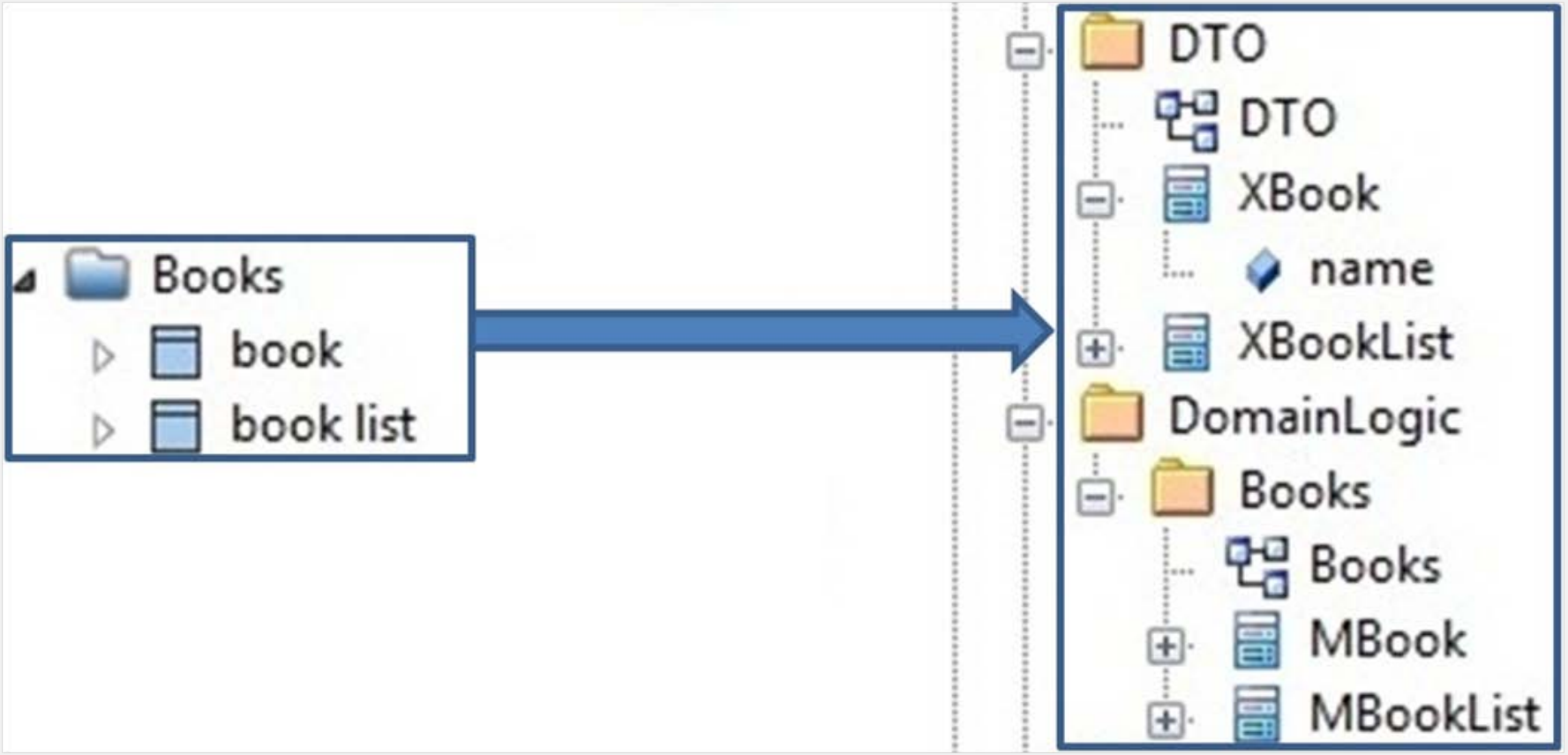
domain logic ⇢



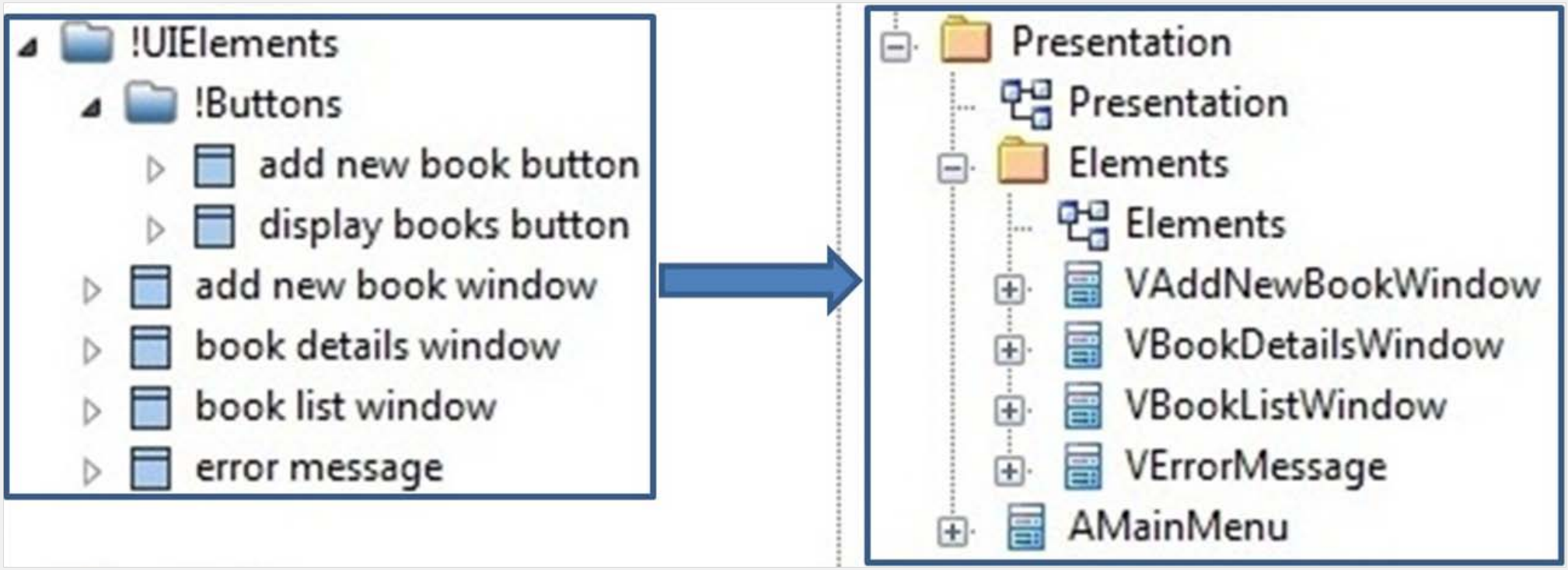
Rule 1: generate the “presenter” classes



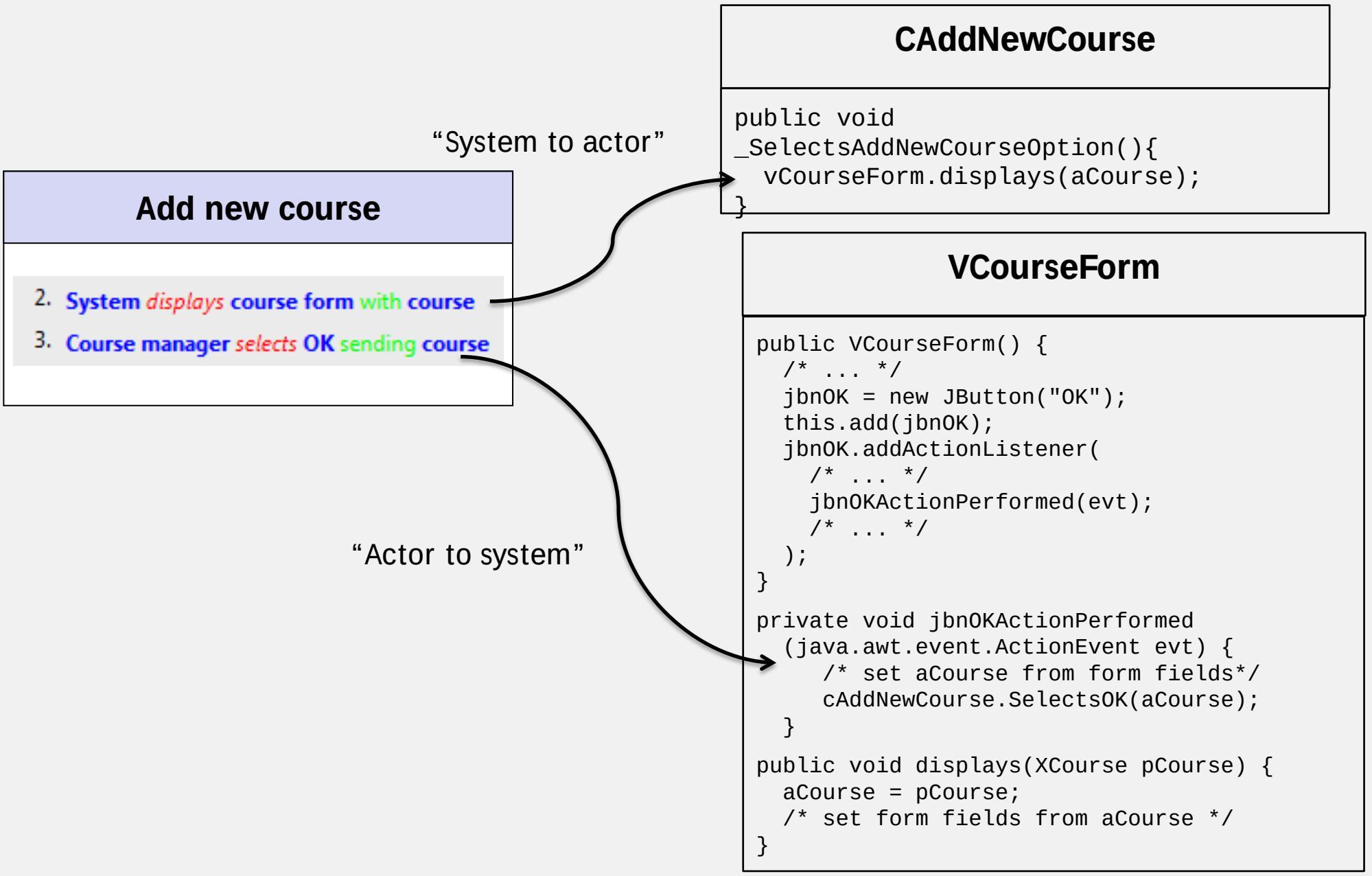
Rule 2: generate the “model” classes & DTOs



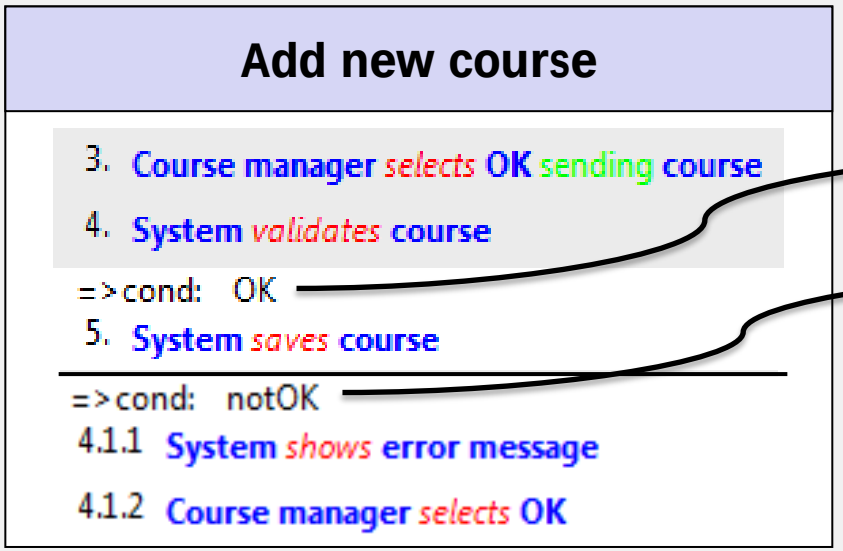
Rule 3: generate the “view” classes



Rules 3 and 4: generate procedure calls



Rules 5 and 6: generate “if”s and procedure calls



CAddNewCourse

```
public void SelectsOK(XCourse pCourse){
  int res;
  aCourse = pCourse;
  mCourse.validates(aCourse);
  res = mCourse.getResult();
  if (res == OK) {
    mCourse.saves(aCourse);
  }
  else if (res == notOK) {
    vErrorMessage.shows();
  }
}

public void SelectsOK_2(){
}
```

MCourse

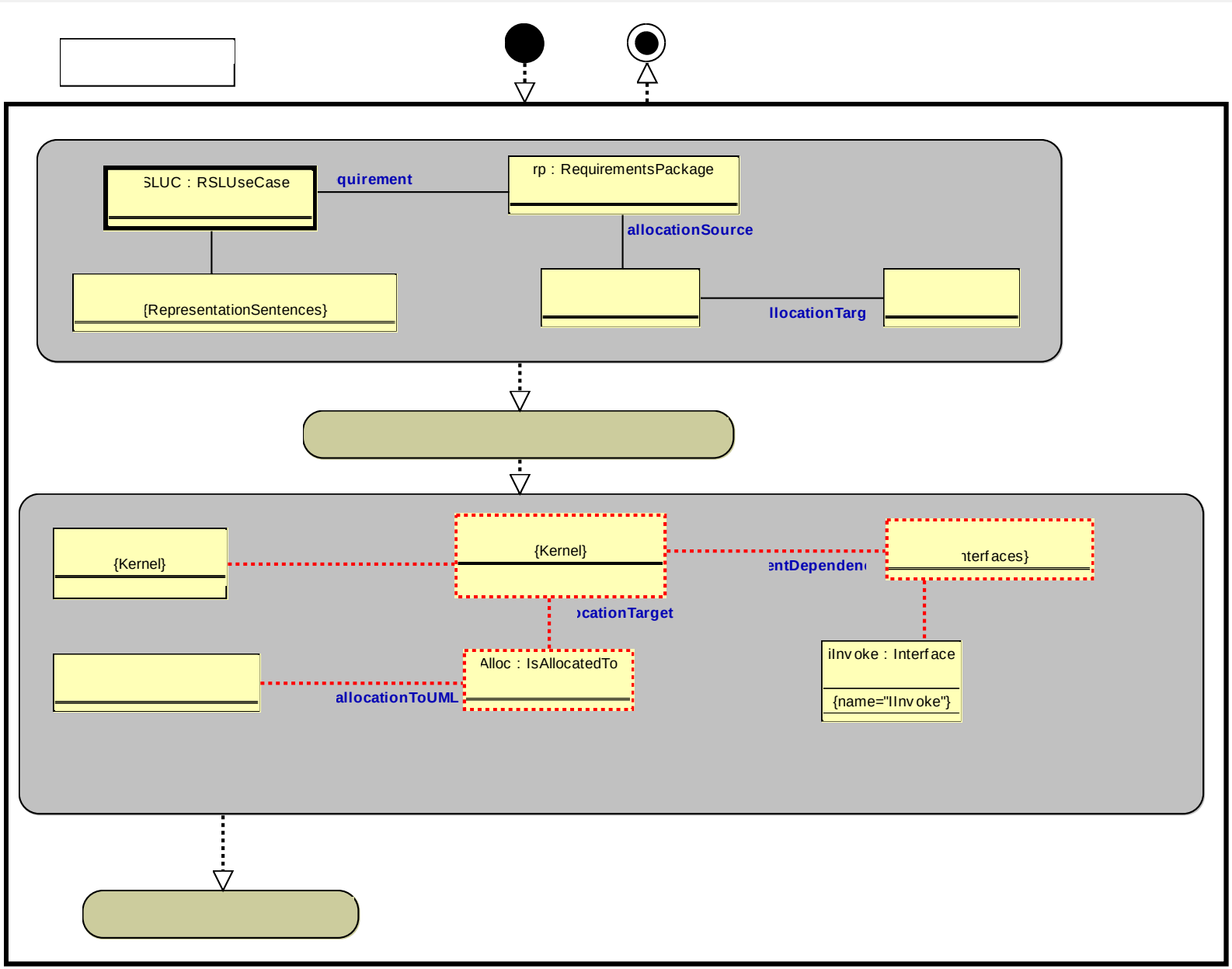
```
public void validates(XCourse pCourse){
}

public void saves(XCourse pCourse){
}
```

Transformation program

Transformation language: MOLA

- MOdel transformation Language
- Graphical language
- Based on metamodel-level pattern matching
- Transformation (program) describes how to transform one model into antoher (RSL to Code)

Example MOLA procedure



Validation and conclusion

- RSL already validated by industry partners (ReDSeeDS Project)
- RSL to Java transformation currently being validated on a banking system (REMICS project)
- Productivity gains validated through student projects
- Significant improvements in productivity, especially for less skilled programmers
- Ready code (“view” and “presenter” parts) can be used directly in production system
- Ready code skeleton available instantly for the data processing (“model”) part

www.remics.eu

www.redseeds.eu

Thank You!