

# Describing Process Patterns with UML

8th European Workshop on  
Software Process Technology

19.6.2001  
Dr. Harald Störrle



## Overview

**1 - Who is FJA? What do we do?**

**2 - Which are our requirements?**

**3 - What is our approach?**

**Examples**

**4 - What are our experiences?**

**5 - Where do we go from here?**



## - Who is FJA?

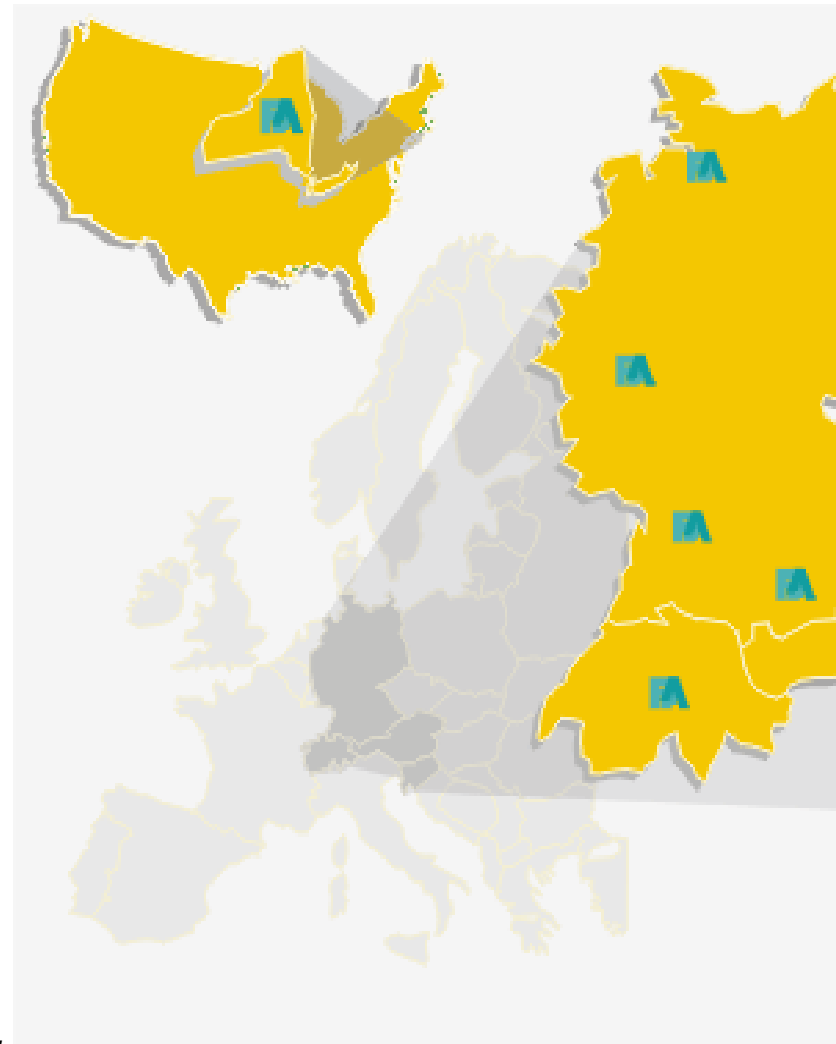
**market leader insurance software**  
*(Germany, Austria, Switzerland)*

**~750 people at 9 offices**  
*(~66% math or CS degree)*

**excellent turnover and profit**  
*(EBIT 2000 ~13.9 Mio. €, + 50%)*

**excellent stock performance**  
*(IPO early 2002: 48 €, now ~72 €)*

**constant growth**  
*(founded 1980, currently ~30% growth p.a.)*



## What does FJA do for a living?

### **Life Factory<sup>®</sup>**

*comprehensive life insurance administration software*

### **Customer Relationship Software**

*rated among best 3 (in our market, for all criteria)*

### **Application Service Providing**

*so called „Riester-Rente“*

...

## - Observations about our Software Processes

### Different kinds of projects...

- ◆ standard software
- ◆ customizing
- ◆ integration
- ◆ ASP

### *...raise different issues*

*architecture, product lines, maintenance  
config. mgmt., time-to-market, training  
coordination of subcontractors, QM  
trouble ticketing, change mgmt., QoS*

### Varying sizes & durations

- ◆ up to 5k person-days for 2-3 years
- ◆ 8 weeks . . . 4 years

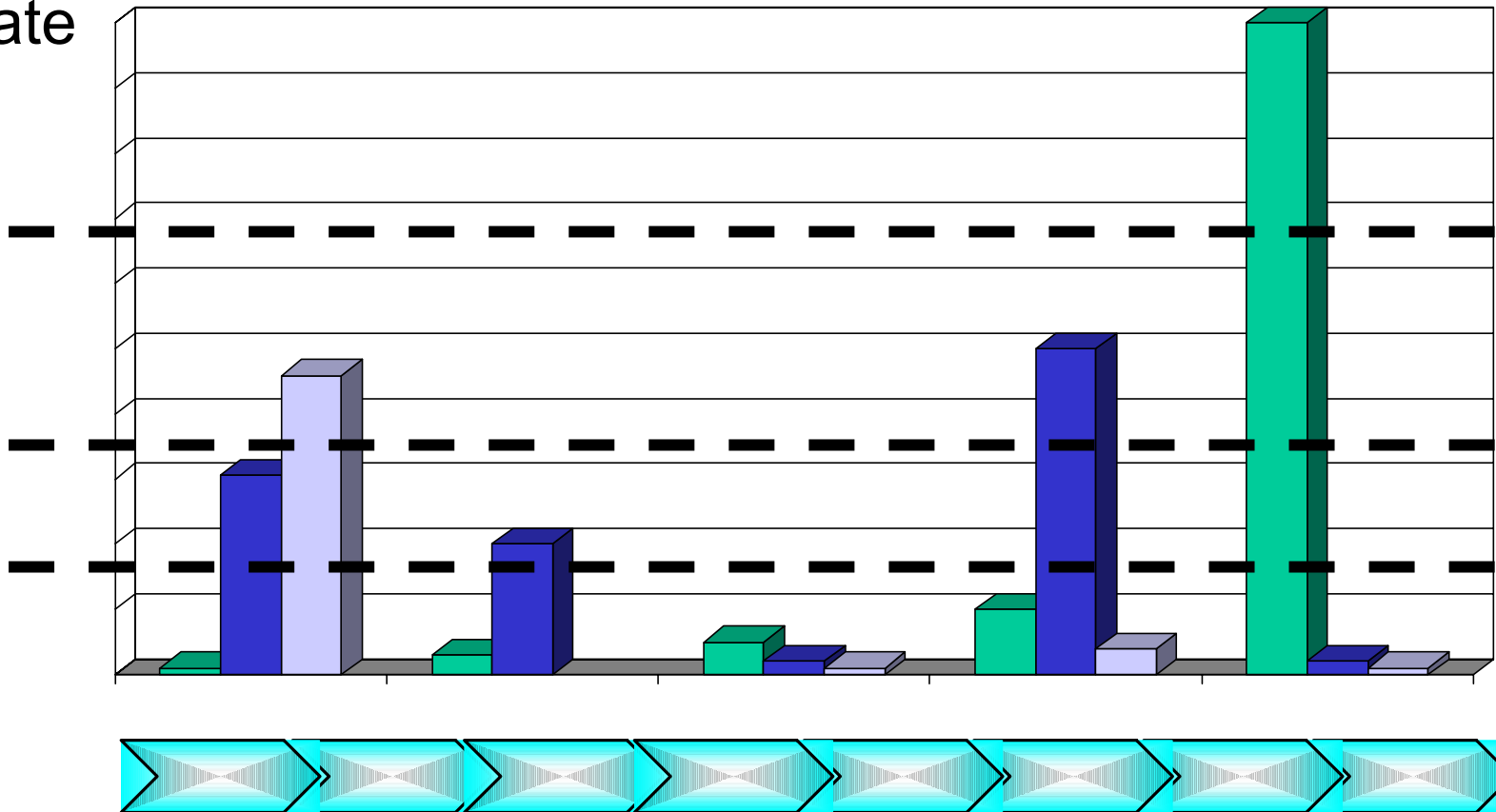
### Risks are spread unevenly



# Risk Profile (fictious)

level of elaboration

work rate



phase



## Example: „Riester-Rente“

**Many insurers need to go into the market *now***

(time-to-market 2-6 months)

**Many non-insurers want to provide such services, too**

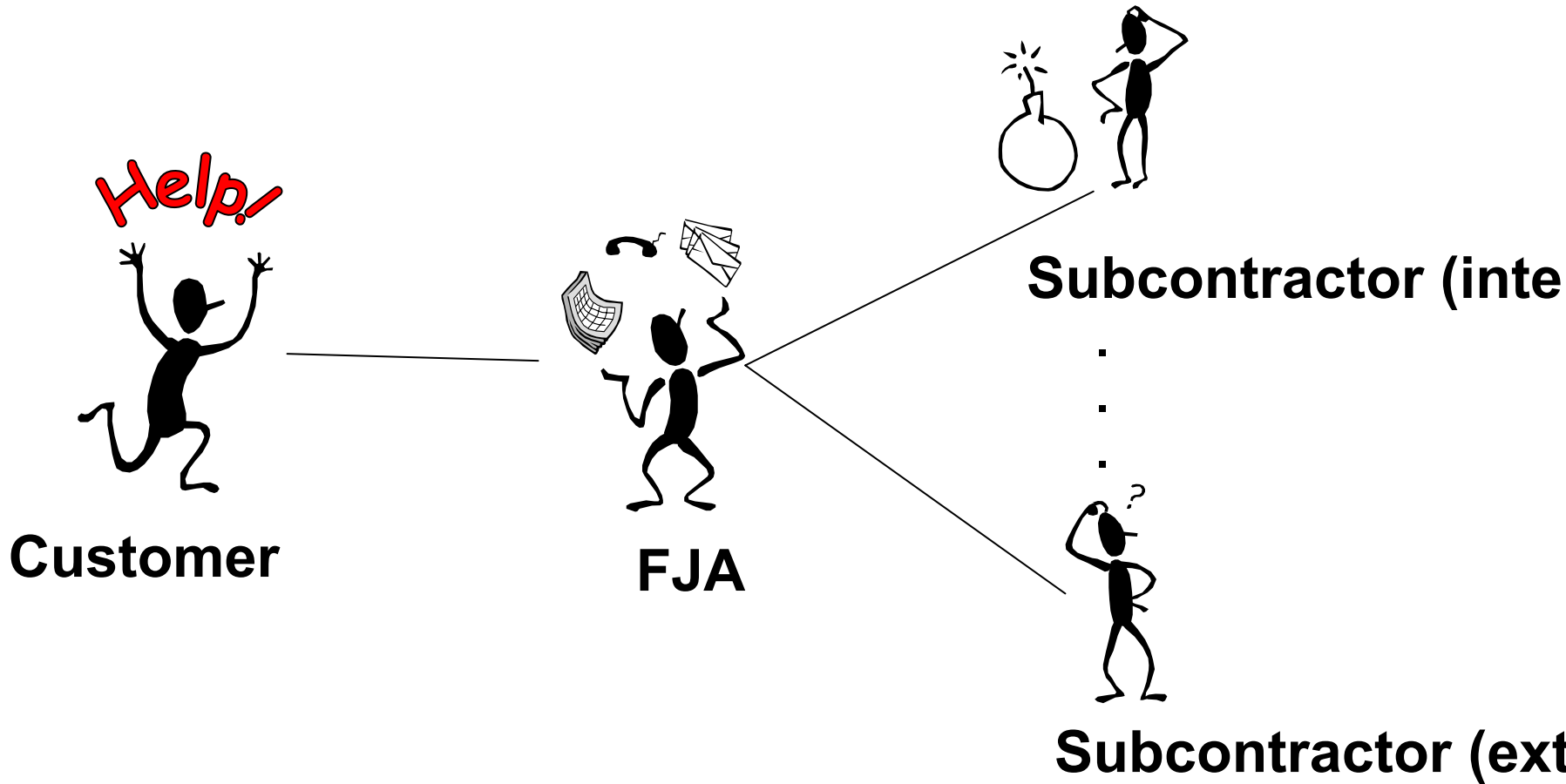
(banks, trade unions, professional bodies, large employers, ...)

### > **Application Service Providing**

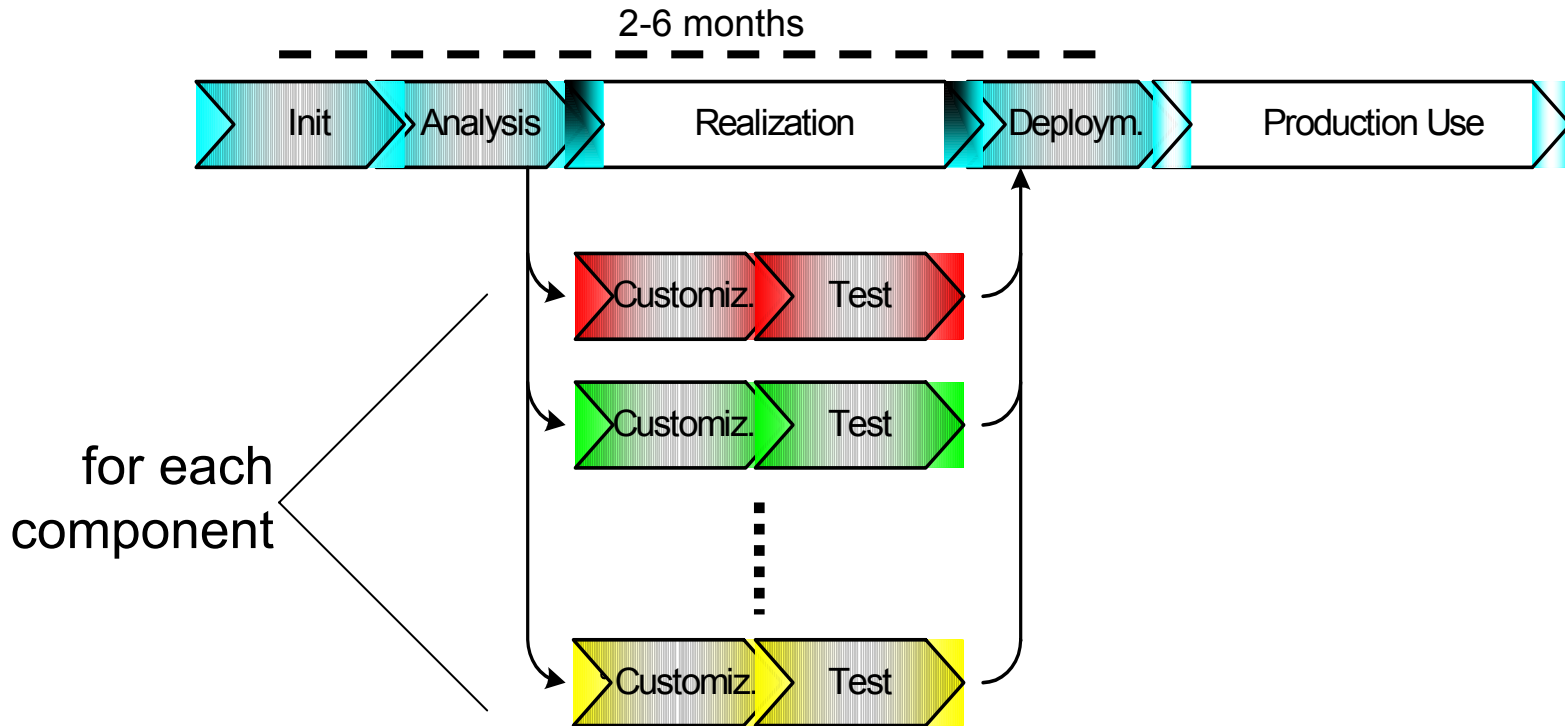
- ◆ integration of standard software / components
- ◆ some integration (customers legacies and partner systems)
- ◆ some customization (beyond switches & DB entries)



# Integration Project Stakeholders



# Integration Project Overall Process



## requirements for integration projects

**Budget is not so important**

**Quality is not so important**

**time to market is the key**



## Possible Process Models (Discarded Alternatives)

**classical models** *(VM'97, RUP, ...)*

- ◆ **overweight & bureaucratic**
- ◆ **hard to learn**
- ◆ **slow reaction to environment changes**

**„lightweight“ models** *(XP)*

- ◆ **doesn't scale, doesn't fit to all kinds of projects**
- ◆ **no handle for process improvement**
- ◆ **customer acceptance (contracts)**
- ◆ **only informal presentations available**

**others?** *(RAD, evolutionary prototyping)*



- Our (current) approach: Process Patterns

***A pattern is a practically proven solution  
to a recurring problem“***

**„one size fits all“** *(incl. component-based development)*

**easy on-the-fly adaptation**

**selective precision: good cost / benefit ratio**

**controlability / improvability**

**best practices oriented** *(both extract and apply)*

**easier to learn**



## How to do it?

**use standard descriptive scheme for patterns**

*(Go4, Go5)*

**use UML for process aspect**

*(RUP-additions plus proper metamodel-embedding)*

**classical terminology remains the same**

*(roles, artifacts, glossary, ...)*

**generate different presentations from XML**



## Example (1/3) - describing a process pattern

**Name**

**Classification**

**Intent**

**Synonyms**

**Motivation**

**Applicability**

**Process**

**Roles**

**Consequences**

**Sample Execution**

**Implementation**

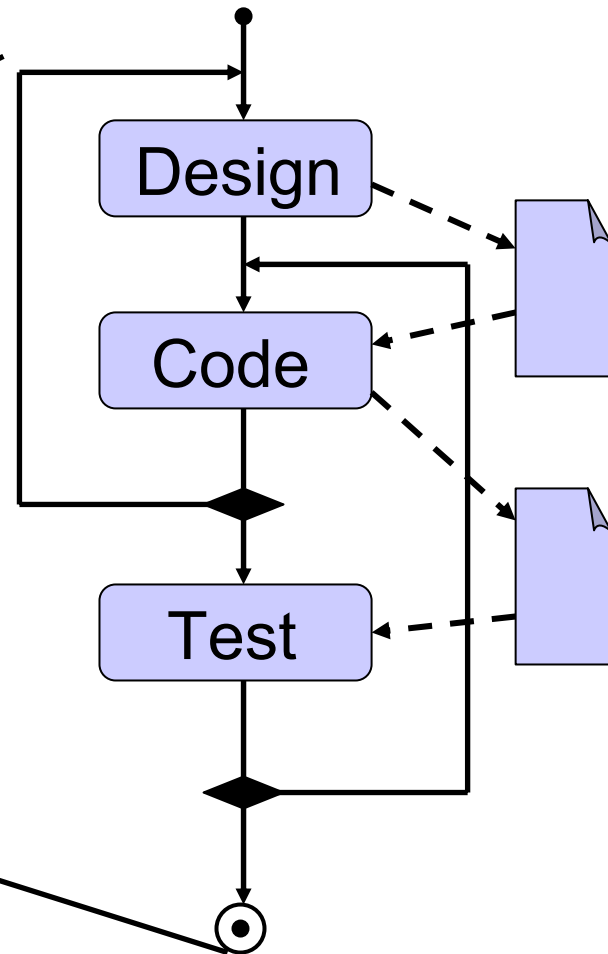
**Known Uses**

**Related Patterns**



# Example (1/3) - describing a process pattern

- Name
- Classification
- Intent
- Synonyms
- Motivation
- Applicability
- Process**
- Roles
- Consequences
- Sample Execution
- Implementation
- Known Uses
- Related Patterns



## Example (1/3) - describing a process pattern

Name

**Classification**

Intent

Synonyms

Motivation

Applicability

Process

Roles

Consequences

Sample Execution

Implementation

Known Uses

Related Patterns

- *abstraction level*
- *technique*
- *activity*
- *process*

- *phase*
- *analysis*
- *design*
- *...*

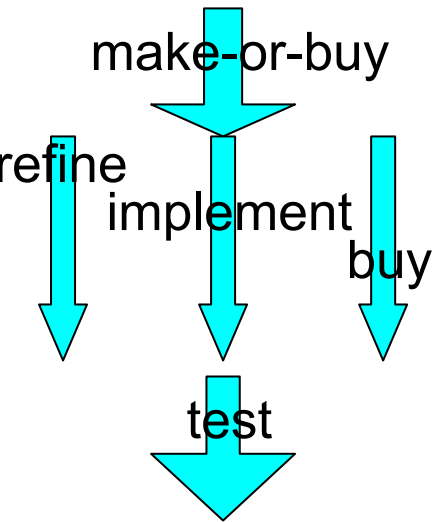
- *Purpose*
- *engineering*
- *support*
- *control*

- *...*

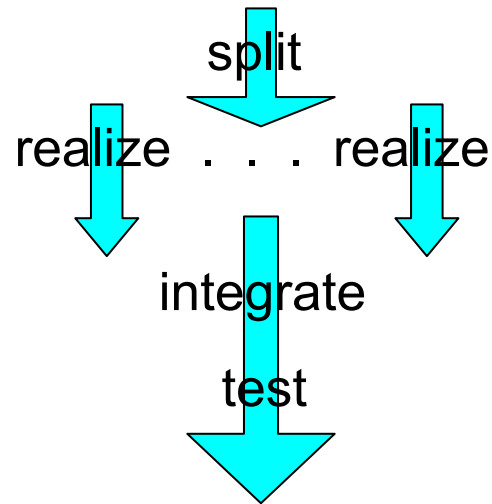


# Example (2/3) - a language of process patterns

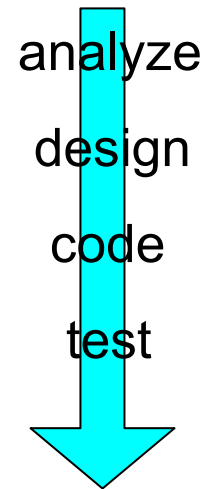
## Realize



## Refine

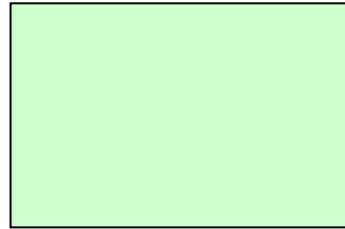
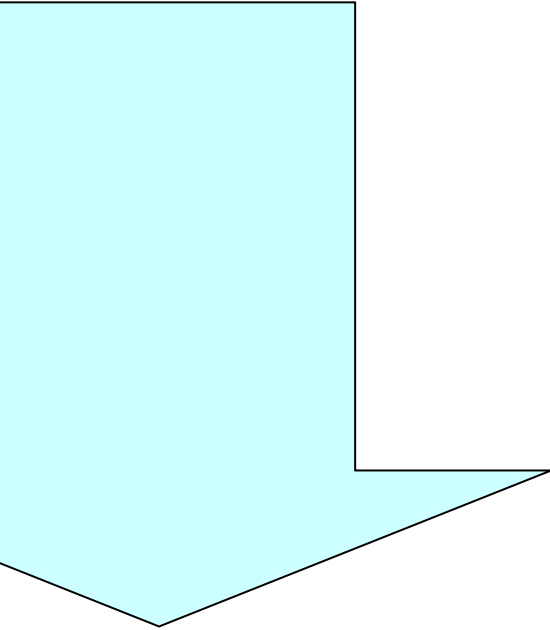


## Implement



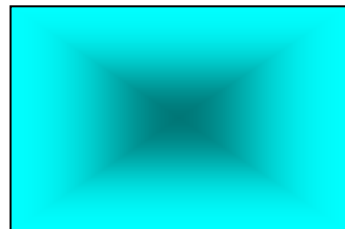
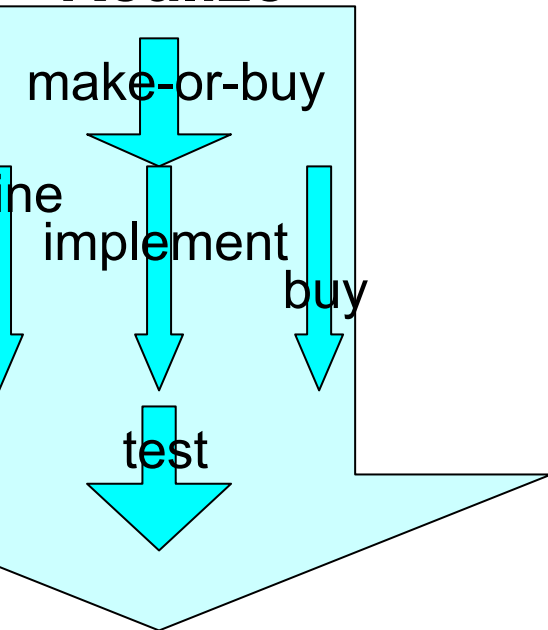
# Example (3/3) - applying process patterns

**Realize**



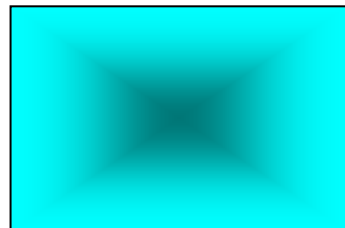
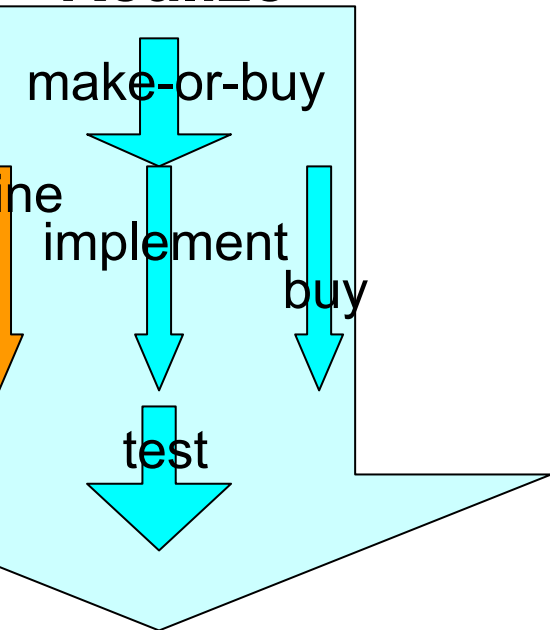
# Example (3/3) - applying process patterns

## Realize

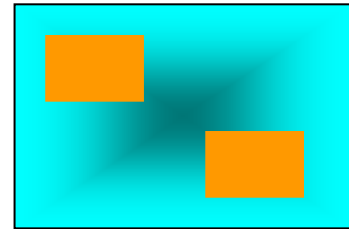
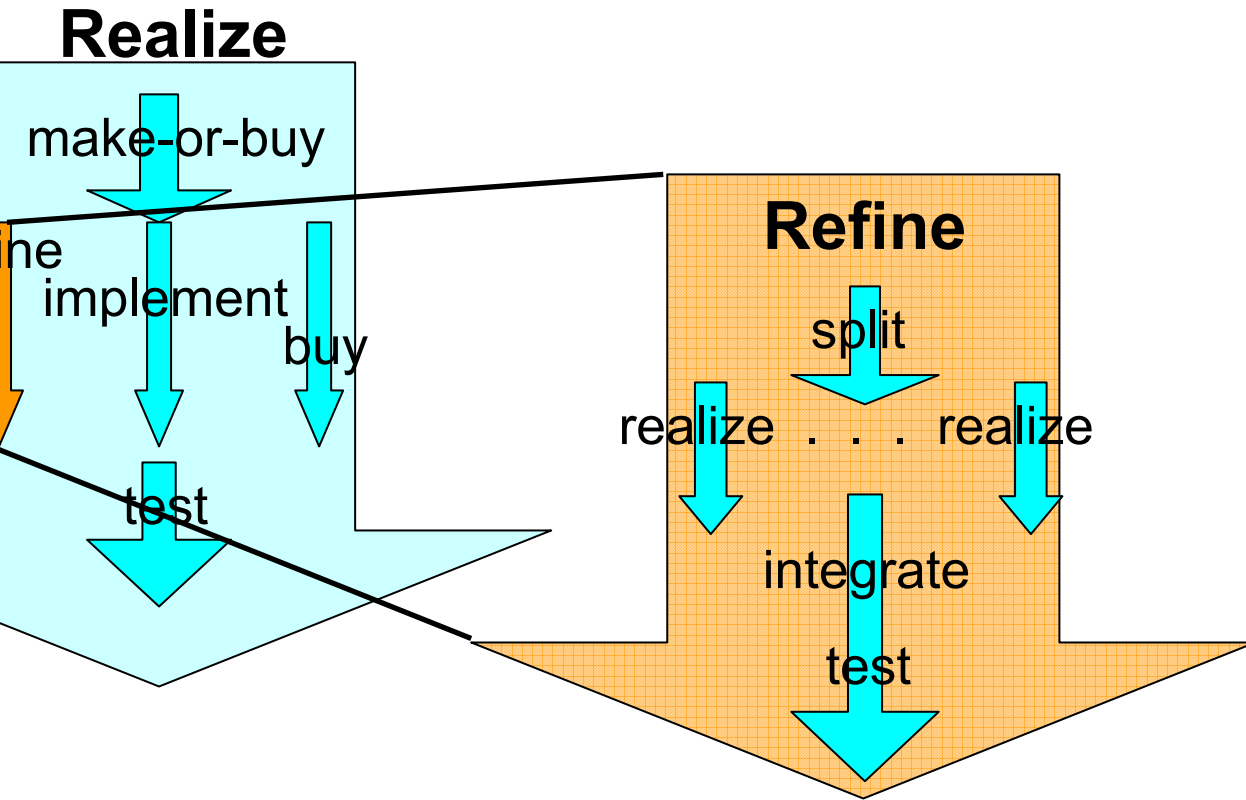


# Example (3/3) - applying process patterns

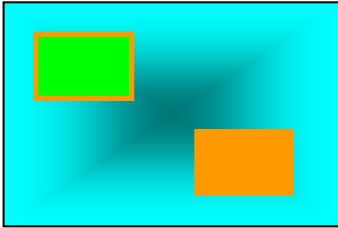
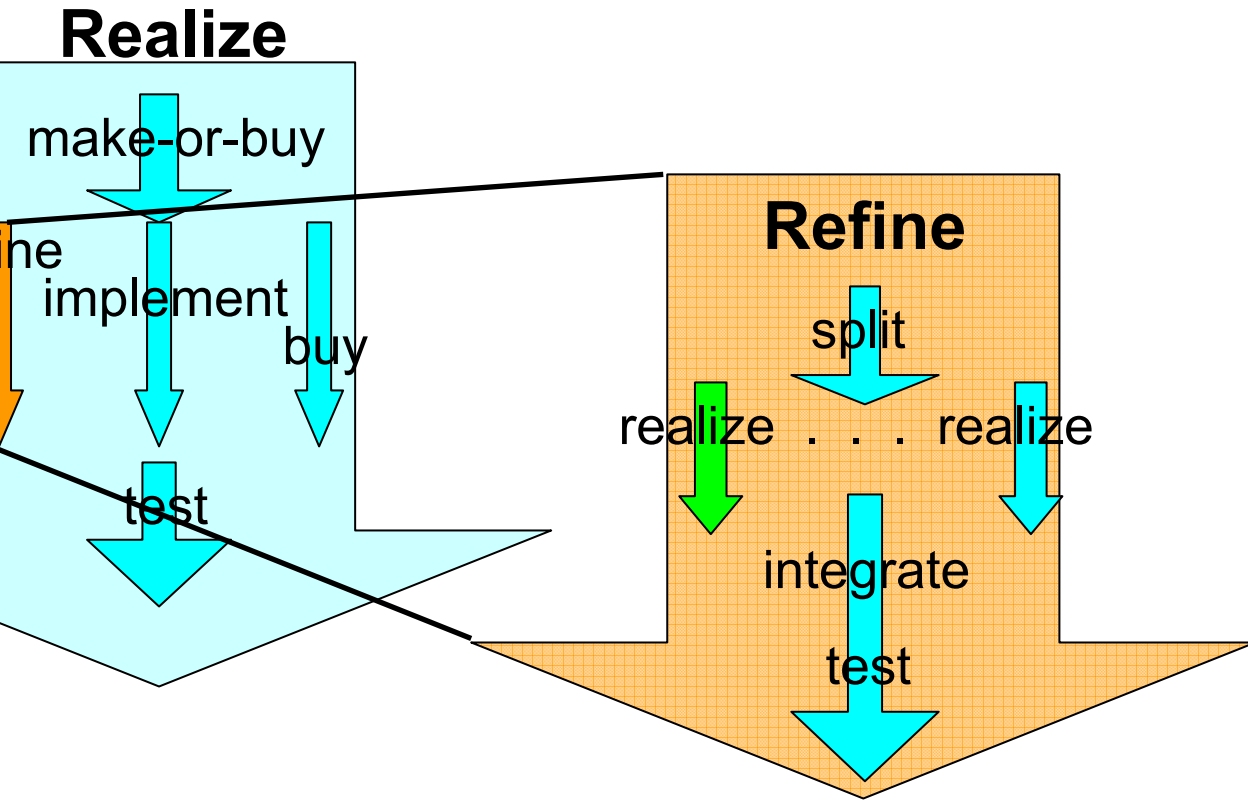
## Realize



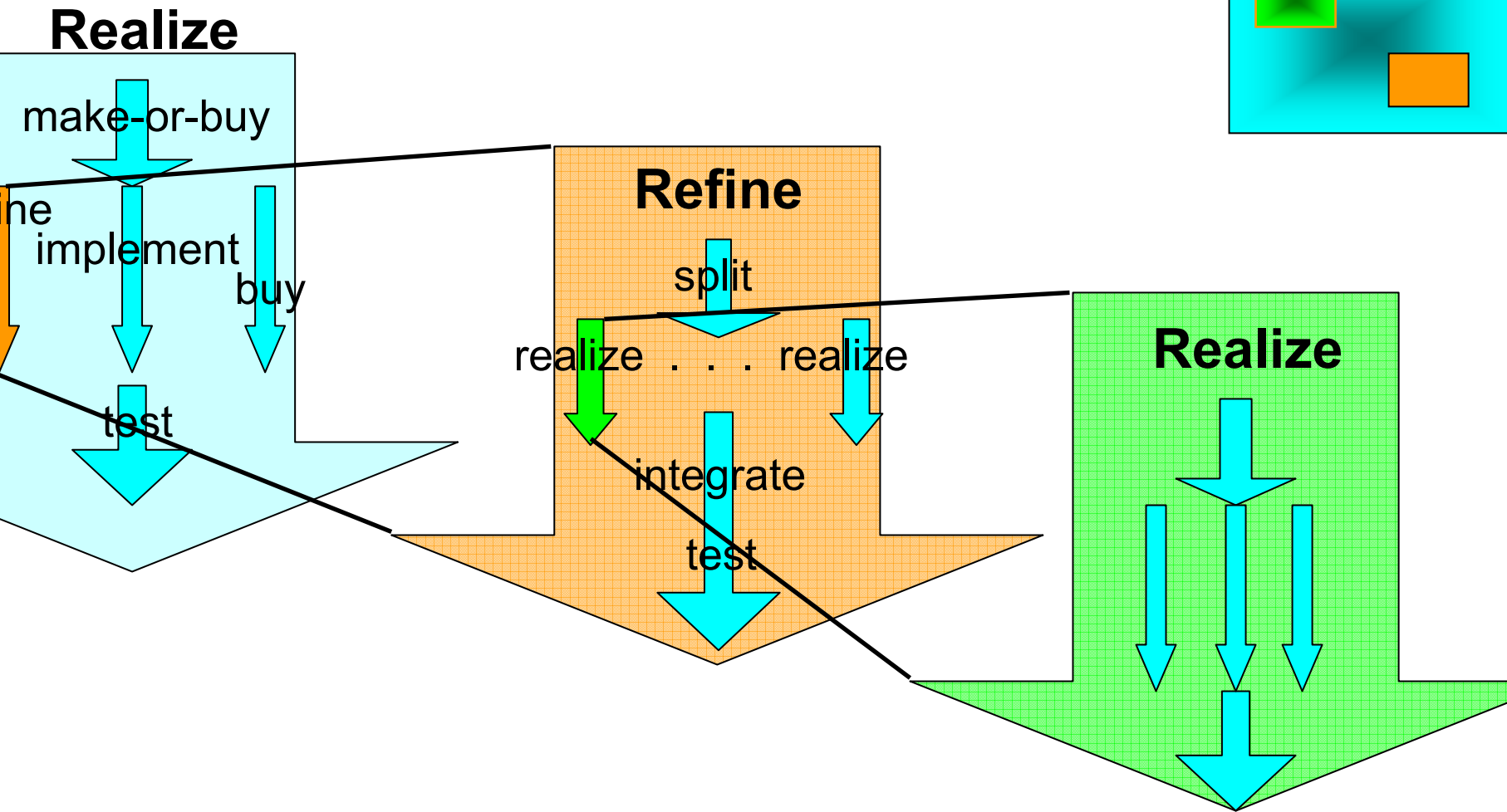
# Example (3/3) - applying process patterns



# Example (3/3) - applying process patterns



# Example (3/3) - applying process patterns



## Consequences

**recursive application of patterns**

*recursive system structure*

**same patterns on all levels**

*scale invariant (fractal) processes*

**easier process implementation**

*simpler, faster, cheaper*



**- What are our experiences?**

**Not yet sufficient practical experience,...**

**but a „beg-borrow-steal“ approach with**

**„implementation by coaching“ seems to be working.**

**Yet, it is difficult to communicate to management**



**- Where do we want to go from here?**

**Future work**

**Other aspects**

**Comments? Questions? Suggestions?**

**=> [Harald.Stoerrle@fja.com](mailto:Harald.Stoerrle@fja.com)**



# Further Issues



## Differences to RUP, VM97

**many detail differences**

**different organization of overall process (concerning activities)**

**adaptive precision**

**all relevant aspects together in same pattern**

(VM: „submodels“, RUP: „core workflows“)

**Similarities: terminology, basics** (roles, artefacts, glossary,...)

VM/RUP are special forms of pattern languages



# Differences to XP

?

?

?

