Assignment
CPPI Optimization vs. PDCA

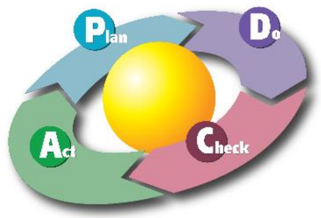
IT-based Management
WS 2015/2016

Thomas.Hiessl@tuwien.ac.at

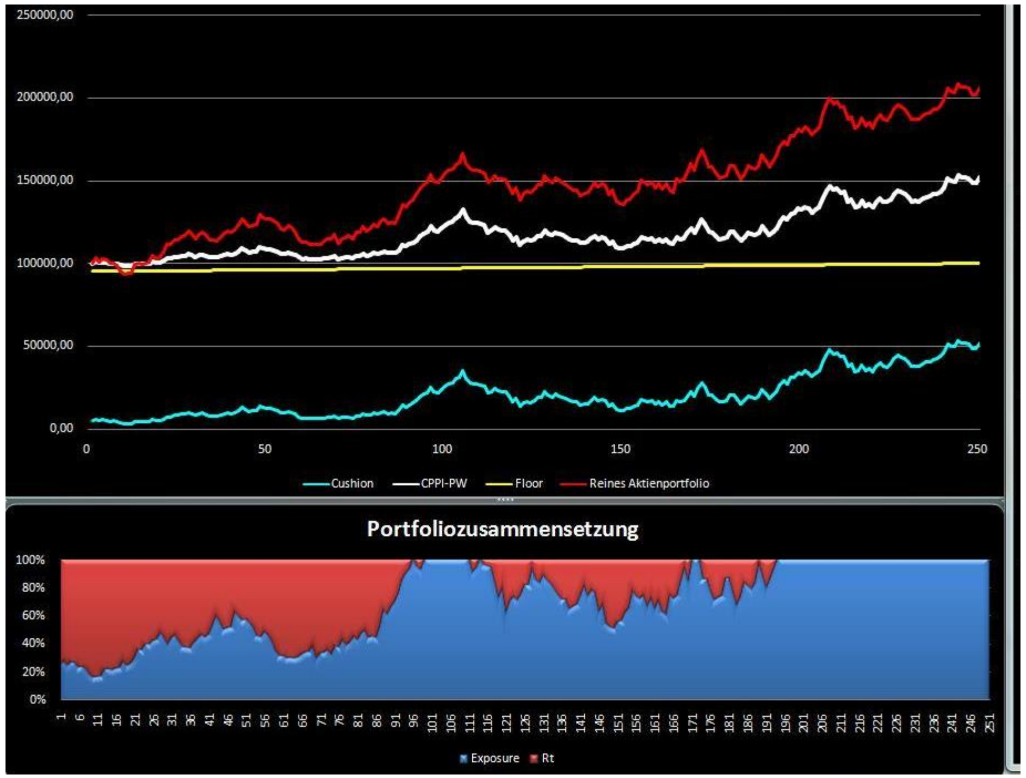
Assignment Definition

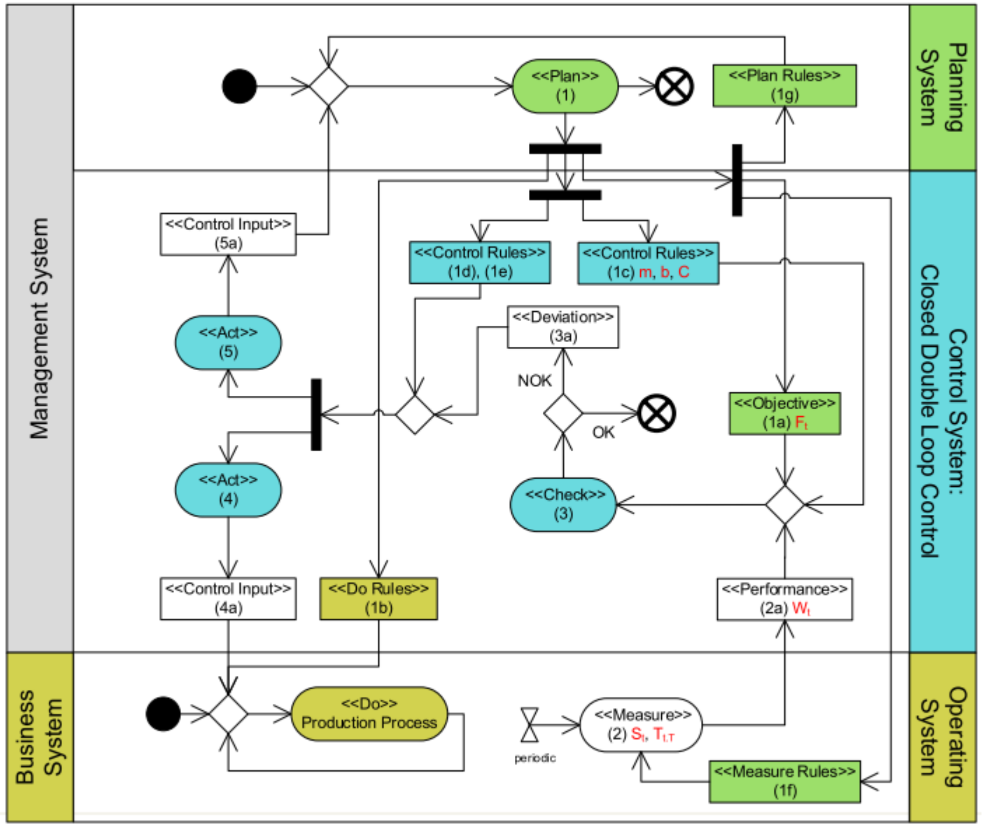
1. Design the CPPI investment strategy with the management activity diagram. **(30%; 6 P)**
   - Choose the right archetype (open/closed/single/double loop) and design the UML activity diagram **(15%; 3 P)**
   - Identify and describe the activities (processes), rules and business objects correctly **(15%; 3 P)**
2. Implement and document a java program, considering a given template, the generic PDCA framework and the designed activity diagram **(70%; 14 P)**
   - Functionality: Print the right CPPI results after each optimization iteration **(35%; 7 P)**
     (Note: input data is provided within the template)
   - Code Quality: Provide a correct implementation of the PDCA framwork **(20%; 4 P)**
   - Documentation: Provide a complete and sound documentation of each activity and rule class **(15%; 3 P)**

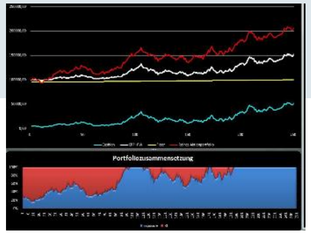
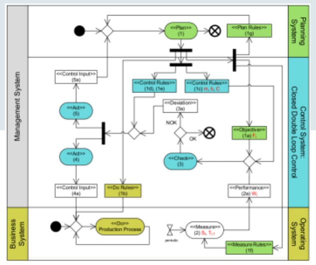
CPPI & PDCA



- Problem statement: Combine the CPPI investment strategy with the PDCA management method





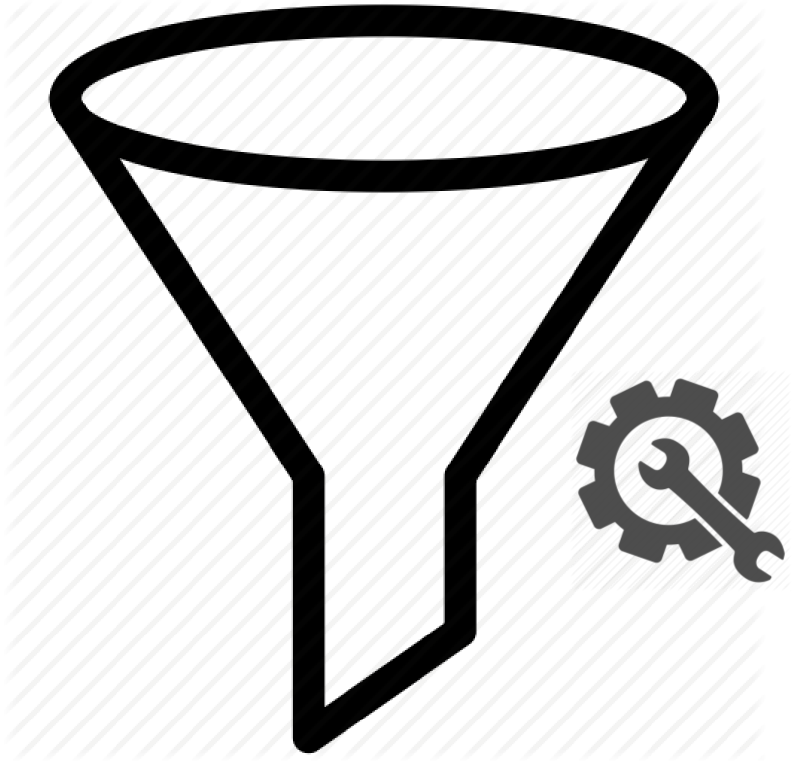
CPPI & PDCA

- Problem statement: Combine the CPPI investment strategy with the PDCA management method
- … by applying concepts of **Financial Engineering:**

1. **Study** the mathematical model of CPPI
2. **Design** a management activity diagram,

   considering the control and information flows
3. **Implement** a java program that optimizes the

   portfolio for the investor

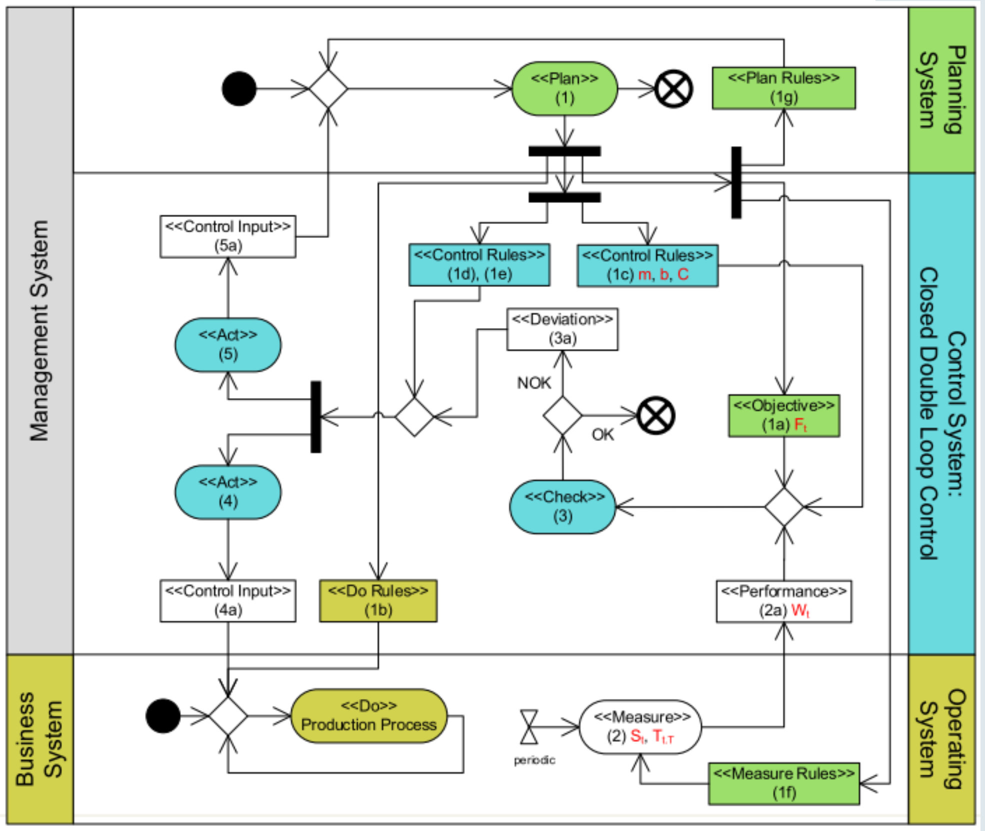


Design I

1. Choose the right management activity diagram configuration:
   - Closed/Single Loop
   - Closed/Double Loop
   - Open/Single Loop
   - Open/Double Loop

   → **Structure**
2. Use the chosen configuration and identify CPPI specific:
   - … Plan- Do- Check- Act- Activities
   - their corresponding rules
   - and their business objects (e.g. <<Performance>>)

   → **Content**



Design II

Hints & Comments:

- Rules:

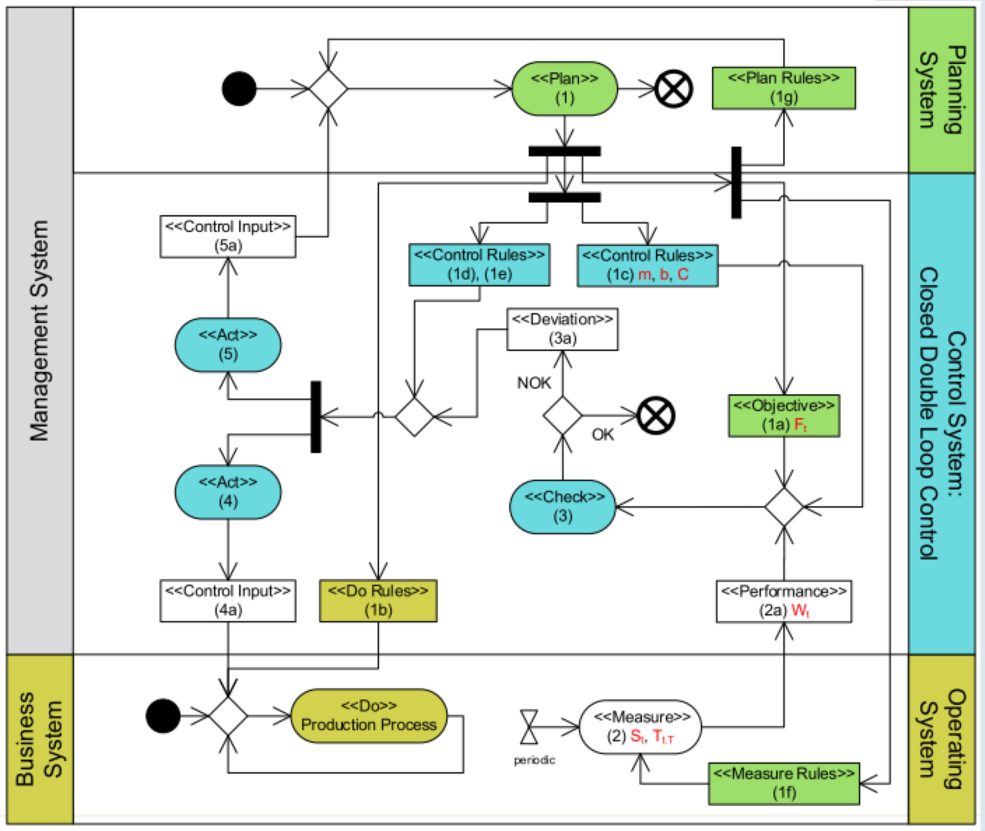
  Determine how to do it. (e.g. CPPI: Cushion calculation equation and if required: parameter)

- Further Business Objects:

  Carry the output of the activity and therefore the input of the succeeding activity

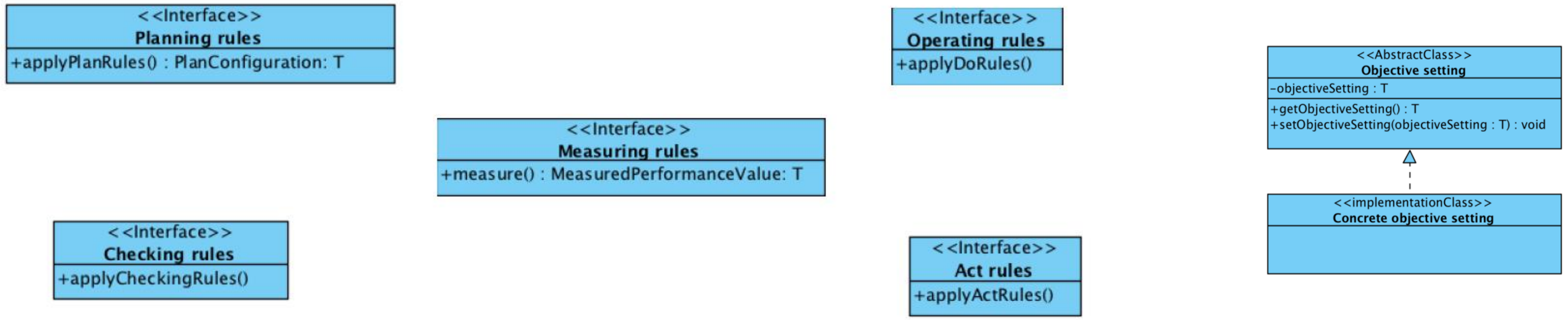
- Activities:

  Apply rules on the current data and produce output



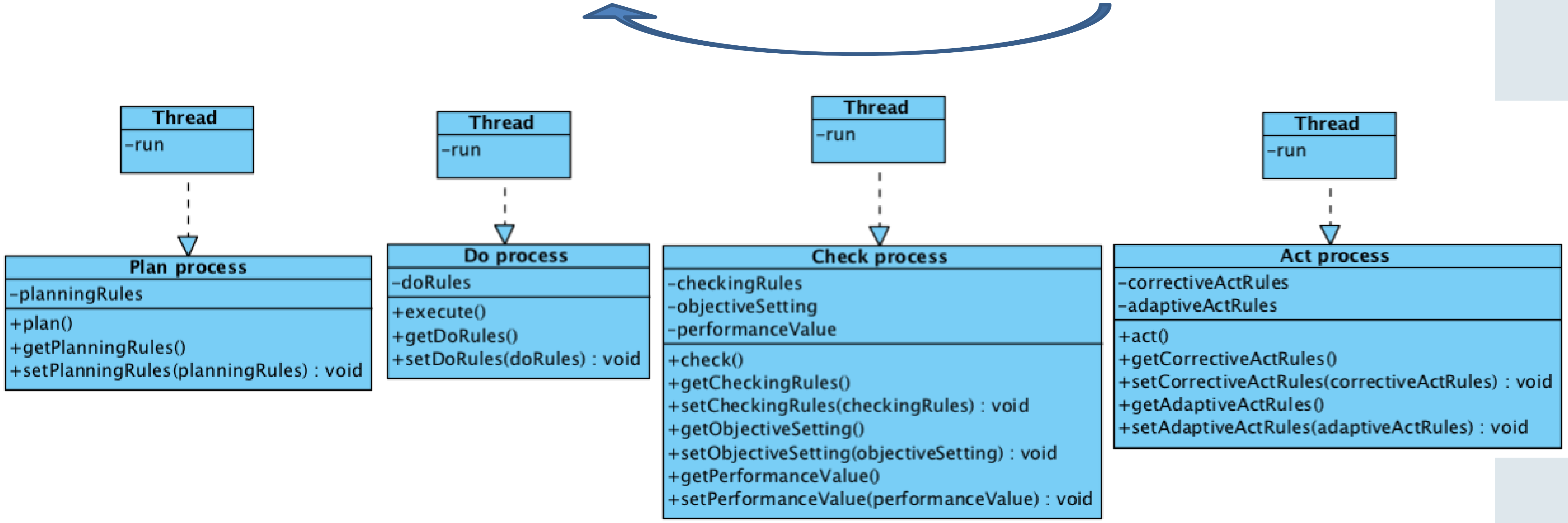
Implementation I

1. Study the generic PDCA framework, which is given to you as a java project
2. Extend and implement the PDCA Rules (Hint: Formulas of the CPPI Model) and other Business Objects (e.g. PlanConfiguration):



Implementation II

3. Extend/Implement the actual PDCA processes
(Hint: apply the rules on the current data , and store the result in a static context object: CPPIService)

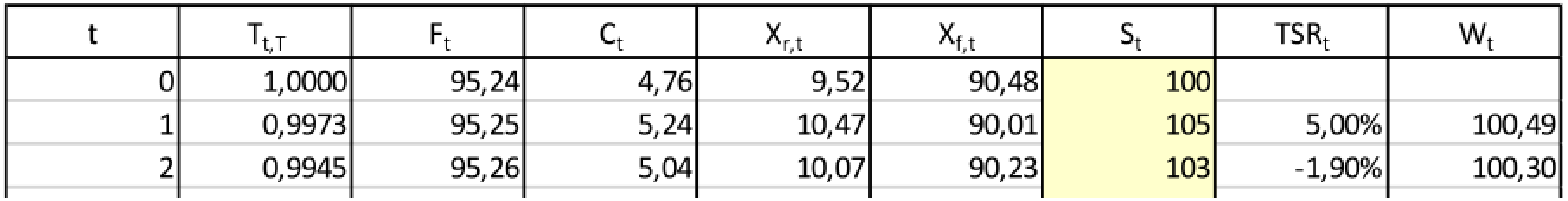


Implementation III

4. Implement a context object (given as CPPIService.java)

   …which contains the current system state (e.g. current portfolio value W^t or measured TSR)

5. Printing one line of result at each t = 1…T:



t	$T_{t,T}$	F_t	C_t	$X_{r,t}$	$X_{f,t}$	S_t	TSR_t	W_t
0	1,0000	95,24	4,76	9,52	90,48	100		
1	0,9973	95,25	5,24	10,47	90,01	105	5,00%	100,49
2	0,9945	95,26	5,04	10,07	90,23	103	-1,90%	100,30

6. Run the processes by:
   - Starting
     - … either each thread (P- D- C- A- M_{easure} –process) separately and let them work by sharing data over the context object
     - Or iterate the PDCAM processes and call them sequentially. (Therefore we sacrifice multi-threading; Hint: might be easier due to the absence of synchronization)

Implementation III – CPPIActProcess.java

```
@Override
public void run() {
    correctiveActRules = … // take from planing output (stored in context object: CPPIService)

    //receive input parameter
    CPPIDeviation deviation = new CPPIDeviation(CPPIService.getInstance().getDeviationValue());

    //set parameter for act rules
    correctiveActRules.setDeviation(deviation);
    //set further parameter here (if needed)…


    //applying the act rule within act(...) and determine new exposure
    correctiveActRules.applyActRules();
    BigDecimal exposure = correctiveActRules.getCorrectiveActOutput().getValue();

    //save results
    CPPIService.getInstance().getCppiValues().setExposure(exposure);
    …
}
```

```
BigDecimal exposure =
deviation.getValue().multiply(leverage).min(rf).multiply(w))
```

… and don‘t forget to log after each iteration, since this is essential for grading! (consider the formatted table from the slide before)

```
public void printLog(int period) {
    log.info(…);
}
```

Comments

- The PDCA Framework is mostly known from social sciences and is therefore applied by humans and rarely by machines but…

- …since we want you to sharpen your PDCA management thinking, this exercise (especially the design part) will contribute to a better understanding of management control.

- …combined with a CPPI, you will be able to understand what „Financial Engineering“ can be like, although there might exist more sophisticated high performance tools for doing this in practice.

- Therefore:
  Don‘t be confused about this mashup → Take the best from both concepts!