

---

---

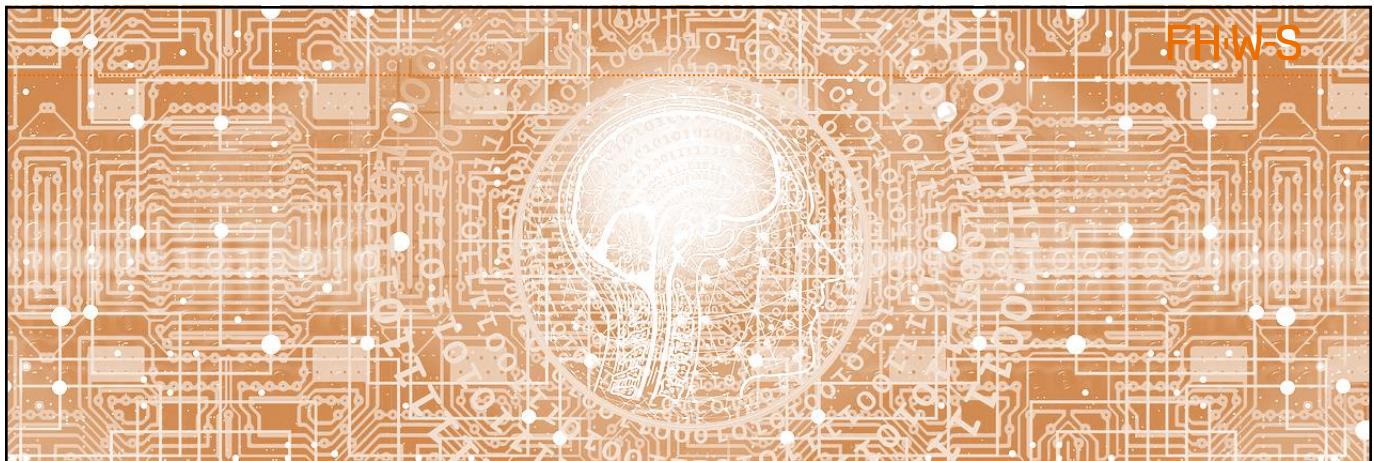
---

---

---







FHWS

# Embedded Systems and Field Buses

**Prof. Dr. Marian Daun**

**FHWS** University of Applied Sciences  
Würzburg-Schweinfurt



Group Discussion

## Expectations

**FHWS** University of Applied Sciences  
Würzburg-Schweinfurt

2



**What this course is about  
and what it is not**

**Fundamentals**  
**Structure of Embedded Systems**  
**Behavior of Embedded Systems**  
**Design of Embedded Systems**  
**Communication**  
**Real-time**  
**Collaborative Embedded Systems**

FH·W-S

# Embedded Systems Fundamentals

FH·W-S University of Applied Sciences  
Würzburg-Schweinfurt

5

# Terminology

FH·W-S University of Applied Sciences  
Würzburg-Schweinfurt

6



Group Discussion

## What is an Embedded System

FH W-S

University of Applied Sciences  
Würzburg-Schweinfurt

7



Exercise

## Define and differentiate from Embedded Systems:

- Information Systems
- Cyber-physical Systems
- IOT-Systems
- Software-intensive Systems
- Safety-critical Systems
- Distributed Systems
- Collaborative Systems

FH W-S

University of Applied Sciences  
Würzburg-Schweinfurt

8

## Characteristics

**Characteristics of Embedded Systems include:**

- **Real-time ability** eg: car brake
- **Safety** – physical devices that can harm people
- **Dependability**
- **Reactivity**
- **Efficiency**
- **New: Security**

## Safety vs Security

In English often used synonymous but when used in German strictly distinguished!

Also officially distinguished according to ISO/IEC/IEEE 24765:

### Safety

- expectation that a system does not, under defined conditions, lead to a state in which human life, health, property, or the environment is endangered

## Safety vs Security

### **Security (according to ISO/IEC/IEEE 24765)**

- protection against intentional subversion or forced failure
- defining, achieving, and maintaining confidentiality, integrity, availability, non-repudiation, accountability, authenticity, and reliability of a system
- degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization
- protection of computer hardware or software from accidental or malicious access, use, modification, destruction, or disclosure
- protection of information and data so that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them
- protection against intentional subversion or forced failure, containing a composite of four attributes: confidentiality, integrity, availability and accountability, plus aspects of a fifth, usability, all of which have the related issue of their assurance



**Group Discussion**

## How do Safety and Security influence each other?

## Dependability

**Typically consists (at least) of:**

- **Availability**
  - degree to which a system or component is operational and accessible when required for use
- **Reliability**
  - ability of a system or component to perform its required functions under stated conditions for a specified period of time
- **Maintainability**
  - ease with which a software system or component can be modified to change or add capabilities, correct faults or defects, improve performance or other attributes, or adapt to a changed environment
  - ease with which a hardware system or component can be retained in, or restored to, a state in which it can perform its required functions
  - ...

[ISO/IEC/IEEE 24765]

## Real-Time

1. **problem, system, or application that is concurrent and has timing constraints whereby incoming events must be processed within a given timeframe**
2. **pertaining to a system or mode of operation in which computation is performed during the actual time that an external process occurs, in order that the computation results can be used to control, monitor, or respond in a timely manner to the external process**

[ISO/IEC/IEEE 24765]

# Reactivity and Efficiency

## Reactivity

- Stimulus and Response
- Input-driven systems

## Efficiency

- degree to which a system or component performs its designated functions with minimum consumption of resources [ISO/IEC/IEEE 24765]
- Hardware is expensive
- Control units are expensive
- Communication is expensive



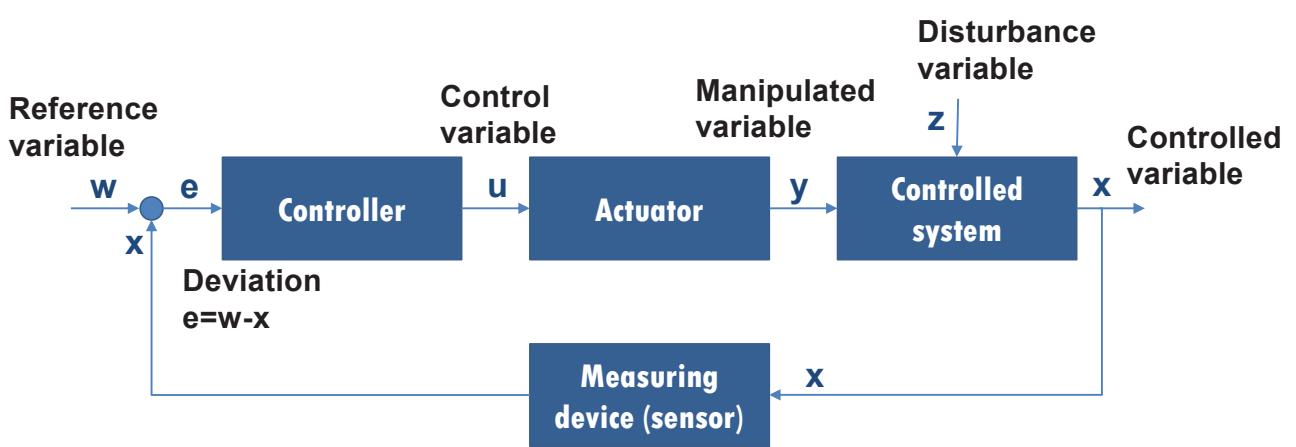
## Exercise

### Where can these characteristics be found in:

- Aircrafts?
- Cars?
- Smart Factories?
- Autonomous Driving?

# Function of Embedded Systems

## Control Loop (Steuer- und Regelkreis)



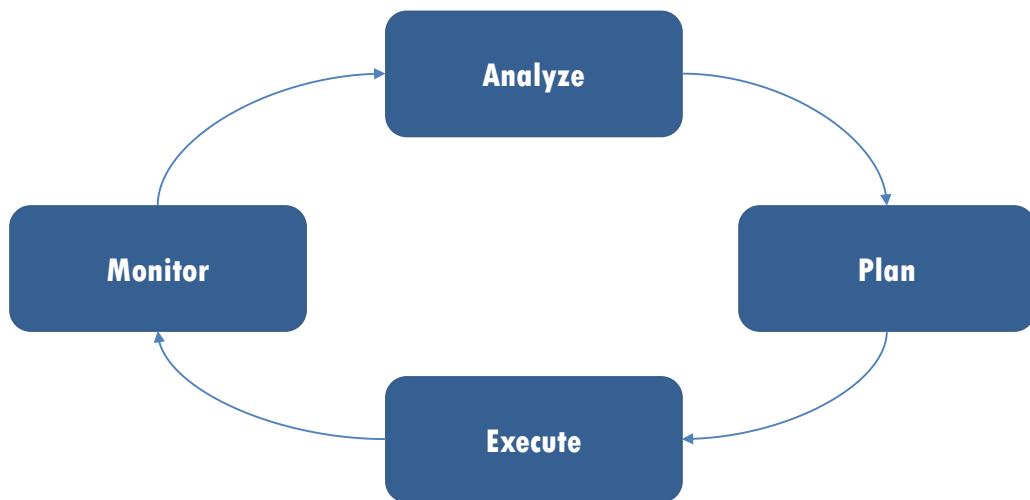


## Exercise

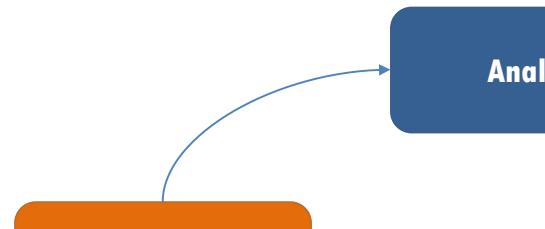
**For each system, identify variables to be controlled:**

- Aircraft
- Car
- Smart Factory
- Autonomous Driving

## MAPE-Loop



## Monitor



**Monitor**

**Be Careful!**

**Values are influenced by multiple MAPE loops!**

**For instance, engine torque is not only increasing when accelerating but, e.g., also when climate control cools down the car.**

**Analyze**

**What values do we need to monitor?**

**What inputs can we actually monitor?**

**How can we calculate/estimate the desired values from the monitored inputs?**

**How often do we need to monitor a value?**

**Examples:**

**Monitor current vehicle's speed via sensor at brakes**

**Monitor engine torque to estimate acceleration**

## Analyze



**Analyze**

**Be Careful!**

**When is a deviation a deviation!**

**Desired speed and current speed will almost never be equal. Define acceptable ranges.**

**Is there a deviation?**

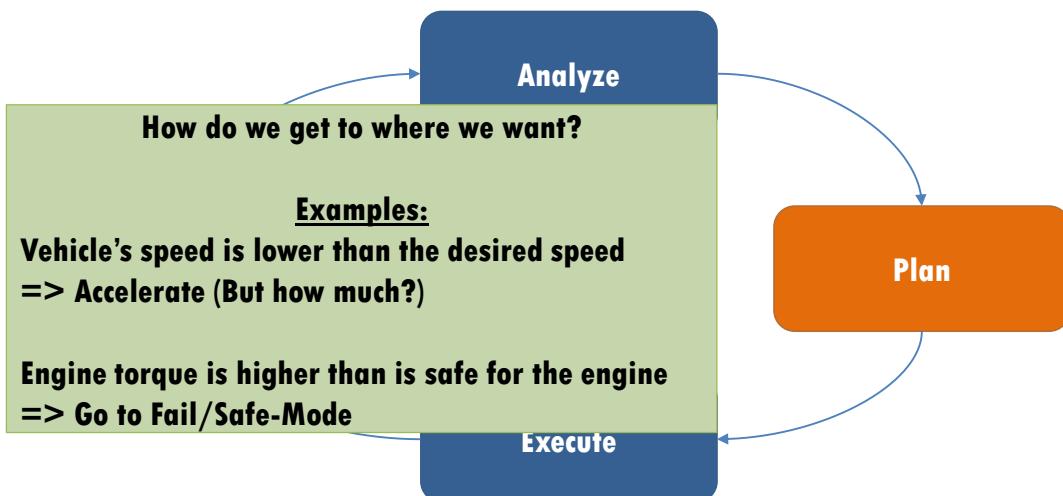
**Examples:**

**Vehicle's speed is lower than the desired speed**

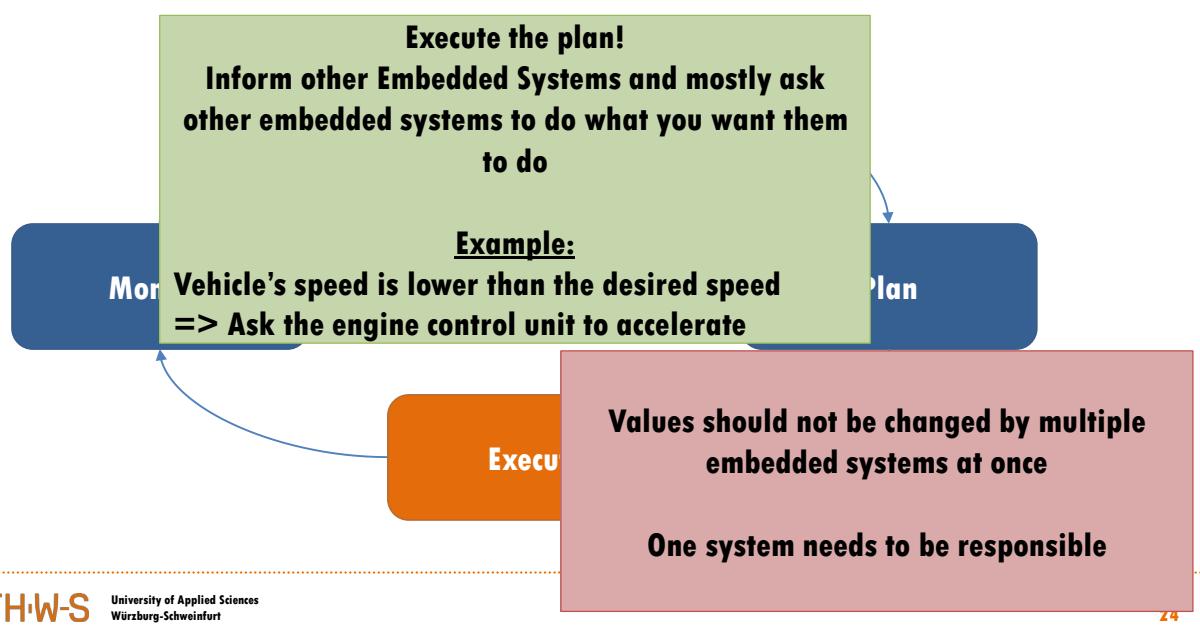
**Engine torque is higher than is safe for the engine**

**Decide**

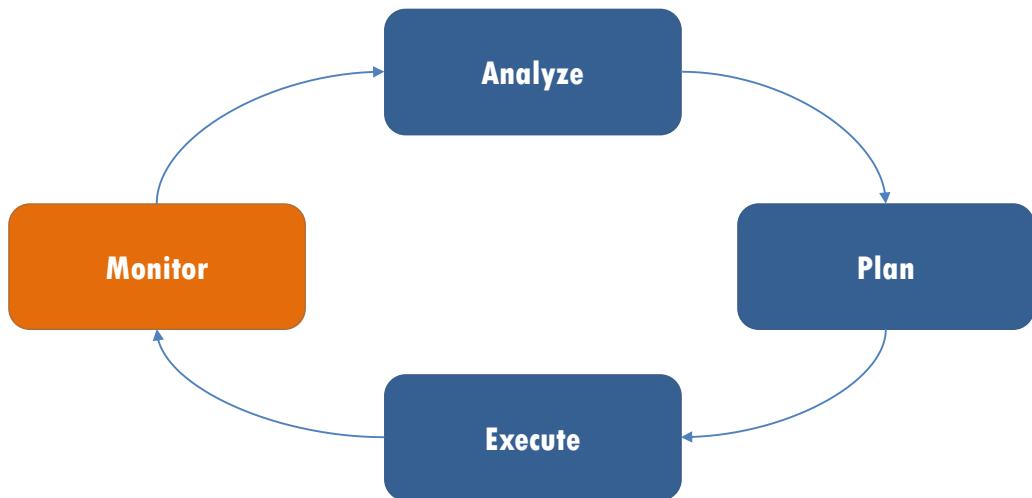
## Plan



## Execute



## And Again: Monitor



**Exercise**

**For each system, define one relevant MAPE-Loop:**

- Aircraft
- Car
- Smart Factory
- Autonomous Driving

## HW/SW Co-design

Engineers like top-down approaches.

However, this does not work with software.

The development of Software and Hardware components of Embedded Systems needs to be intertwined.

There is a multitude of solutions. We can have hardware solutions, software solutions, or mixtures. Thus, the hardware defines what software we need and the software defines what hardware we need.

## Continuous vs Discrete Values

Hardware is continuous

Software is discrete

What does this mean?

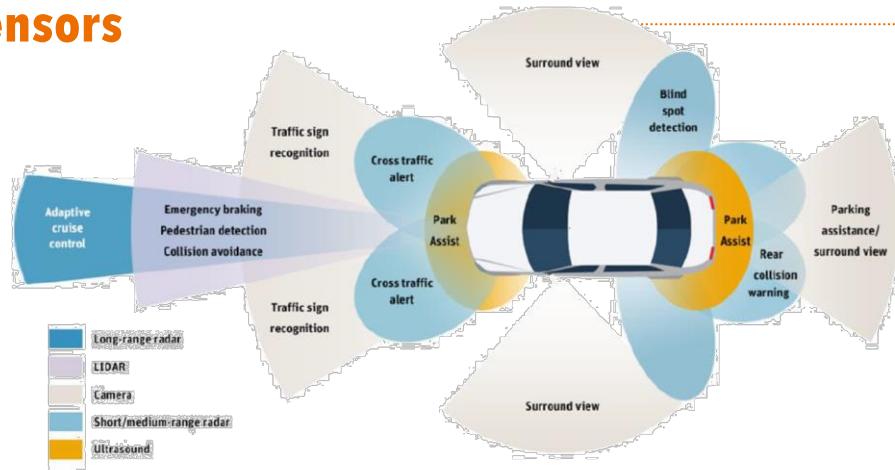
# Sensors



Group Discussion

## What sensors do we know?

## Exemplary Sensors



Slide taken from Lecture  
Materials provided by  
Prof. Dr. Marco Schmidt

<https://www.ansys.com/about-ansys/advantage-magazine/volume-xii-issue-1-2018/autonomous-vehicle-radar>

## Actuators



Group Discussion

## What actuators do we know?



Exercise

For the MAPE-Loops identified before, define the necessary Sensors and Actuators

## Literature

[ISO/IEC/IEEE 24765]

**Systems and software engineering – Vocabulary. International Standard, ISO/IEC/IEEE, 2017.**

FHWS

University of Applied Sciences  
Würzburg-Schweinfurt

35

## Questions for Self-Assessment

**What is an embedded system?**

**How do embedded systems differ from other systems?**

**What are the main characteristics of embedded systems?**

**Where is the difference between safety and security?**

**What is the difference between a control loop and a MAPE loop?**

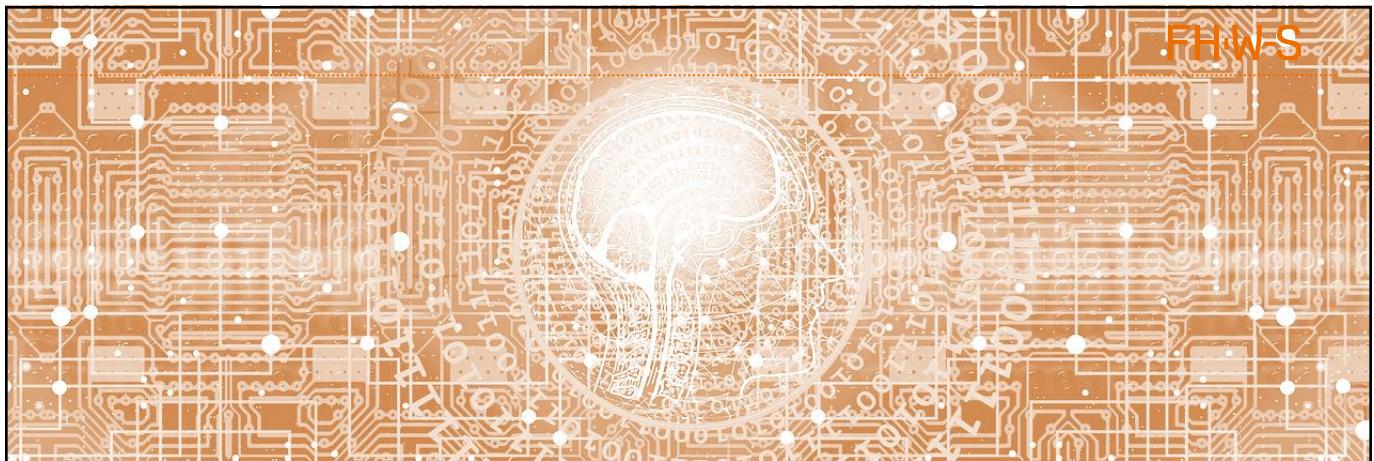
**What problems can arise when defining a MAPE loop?**

**Why is the development of hardware and software intertwined?**

FHWS

University of Applied Sciences  
Würzburg-Schweinfurt

36



# Embedded Systems and Field Buses

Prof. Dr. Marian Daun

**FHWS** University of Applied Sciences  
Würzburg-Schweinfurt

## Agenda

- Fundamentals
- Structure of Embedded Systems
- Behavior of Embedded Systems
- Design of Embedded Systems
- Communication
- Real-time
- Collaborative Embedded Systems

**FHWS** University of Applied Sciences  
Würzburg-Schweinfurt

 FHWS

# Structure of Embedded Systems

 FHWS University of Applied Sciences  
Würzburg-Schweinfurt

3

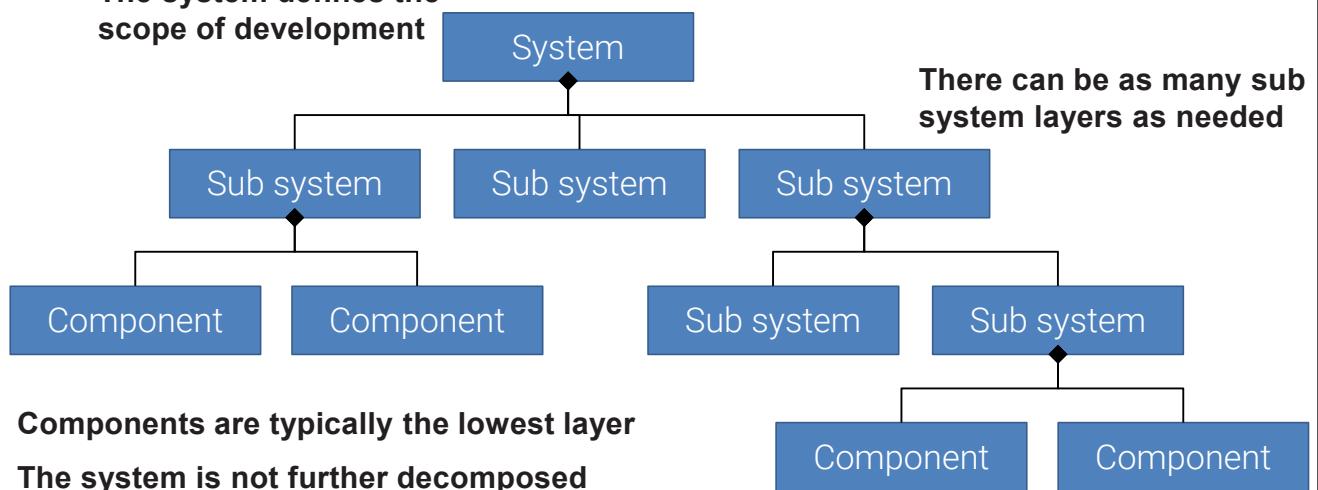
# Hierarchies

 FHWS University of Applied Sciences  
Würzburg-Schweinfurt

4

# System Layers

**The system defines the scope of development**



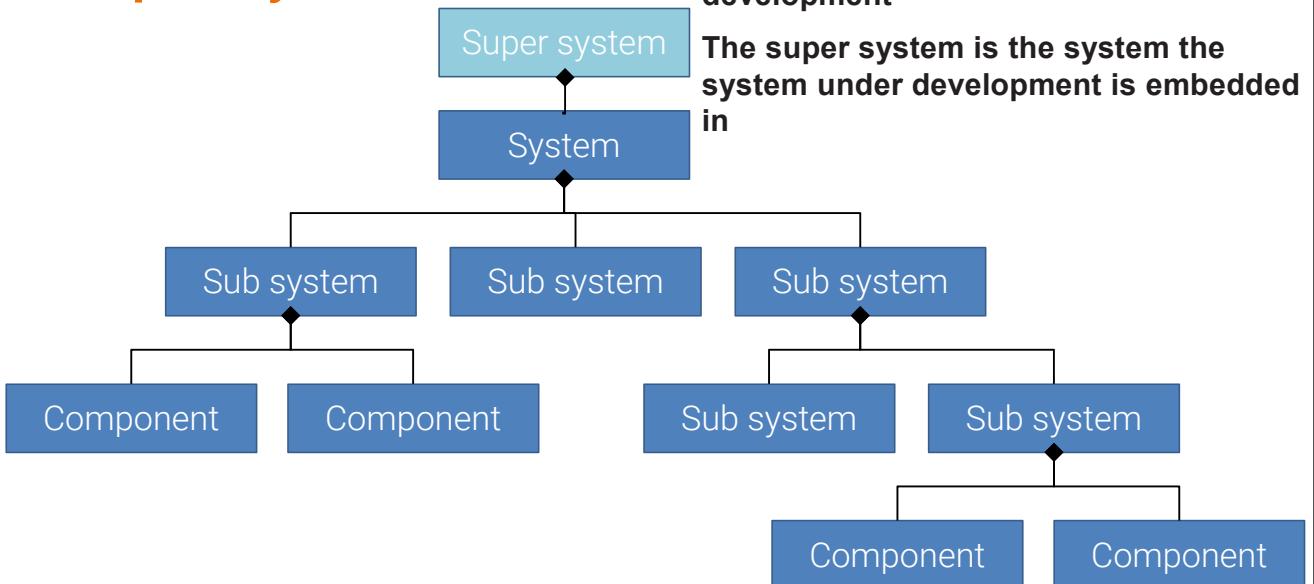
Group Discussion

## When has Development reached the Component Layer?

## Super System

The super system is not the scope of development

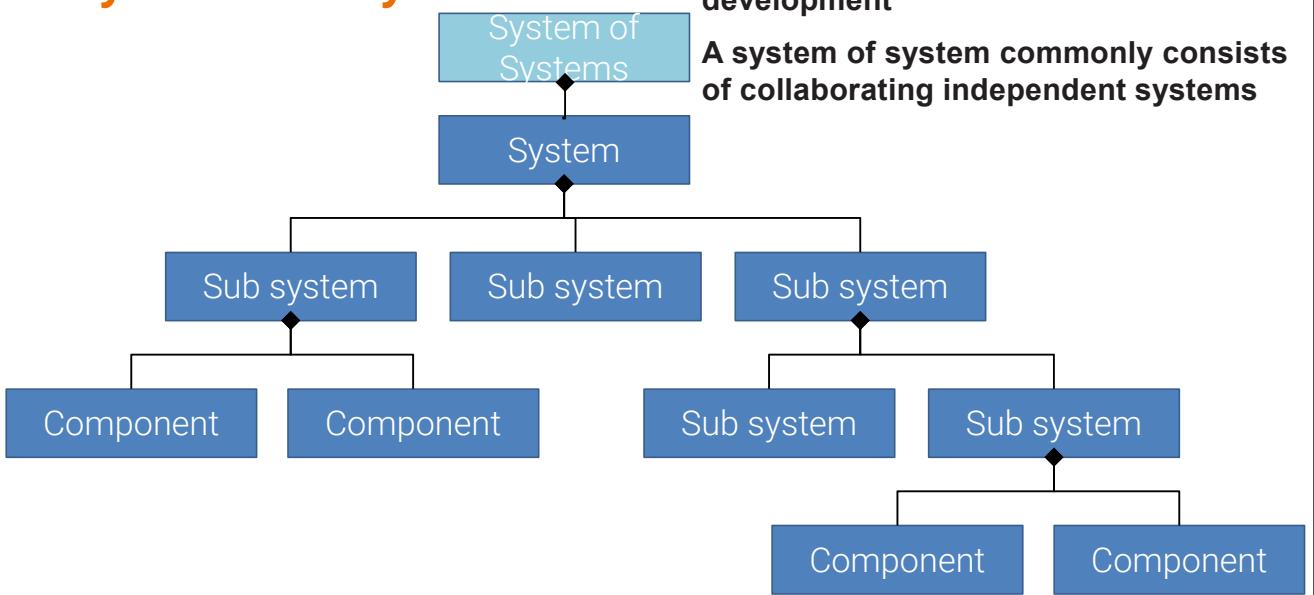
The super system is the system the system under development is embedded in



## System of Systems

A system of systems can be the scope of development

A system of system commonly consists of collaborating independent systems





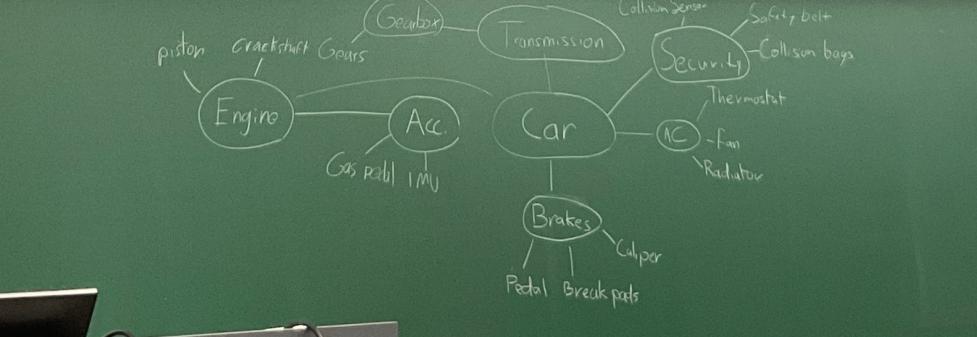
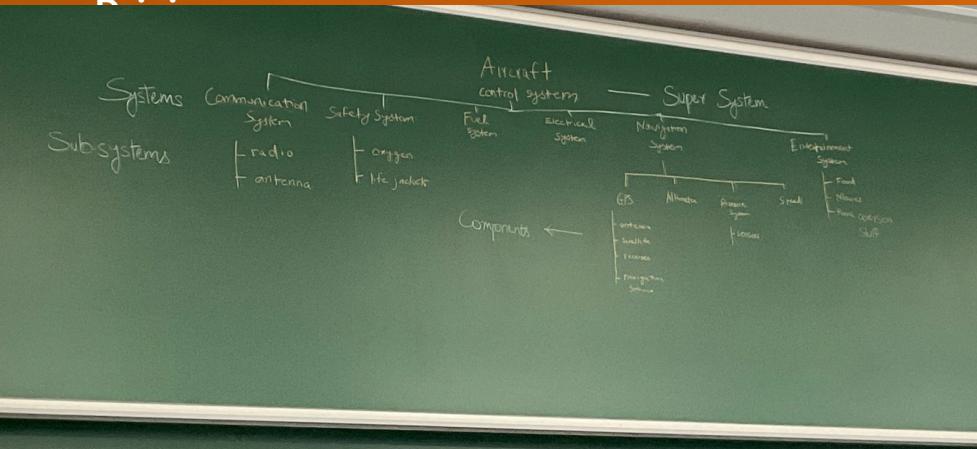
## Exercise

### Define the System Layers for:

- An Aircrafts
- A Car
- A Smart Factory
- Autonom

FH W-S  
University of Applied Sciences  
Würzburg-Schweinfurt

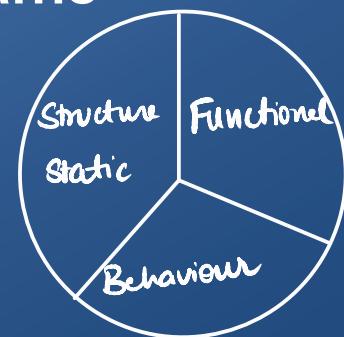
9

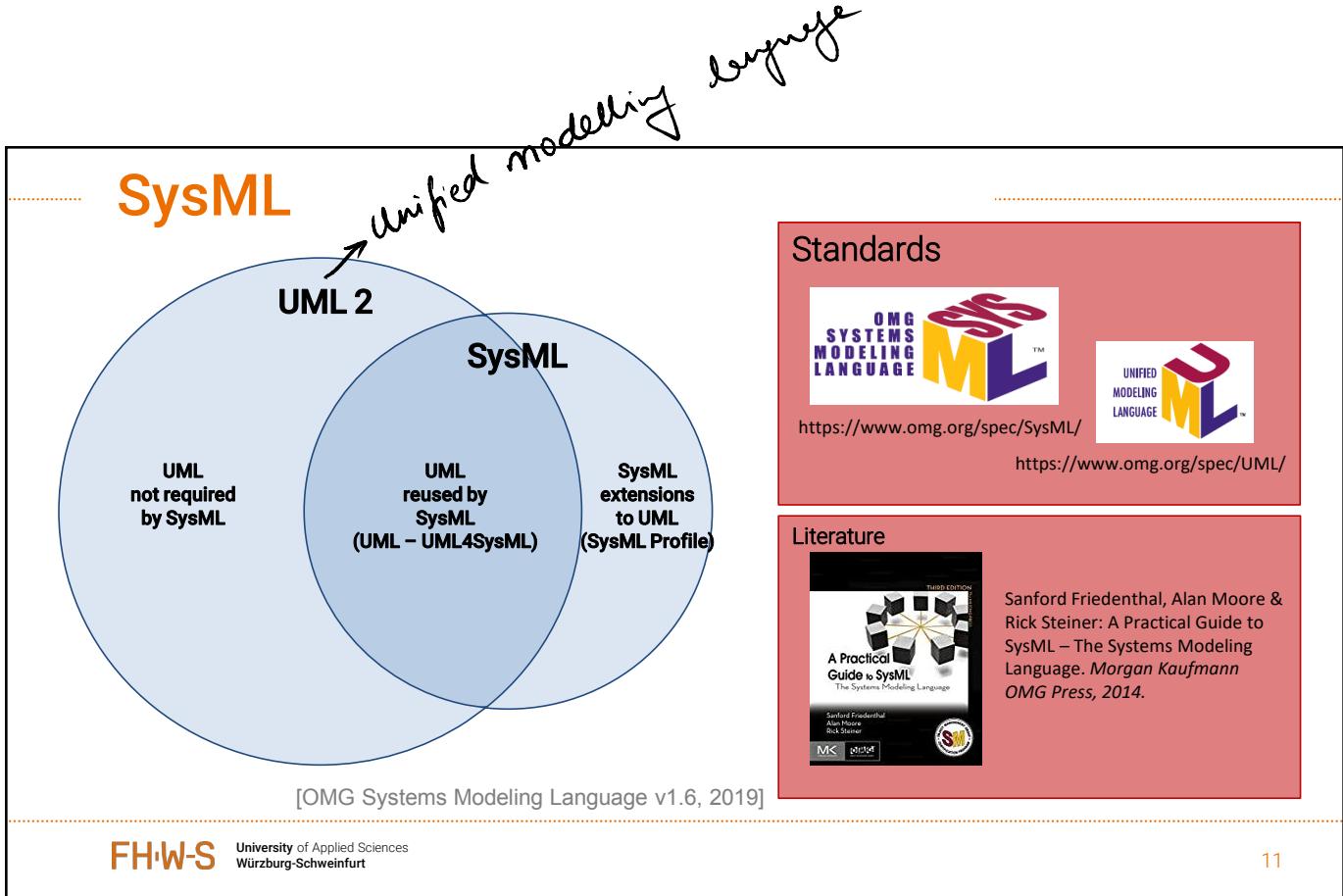


### Modeling with SysML Block Definition Diagrams

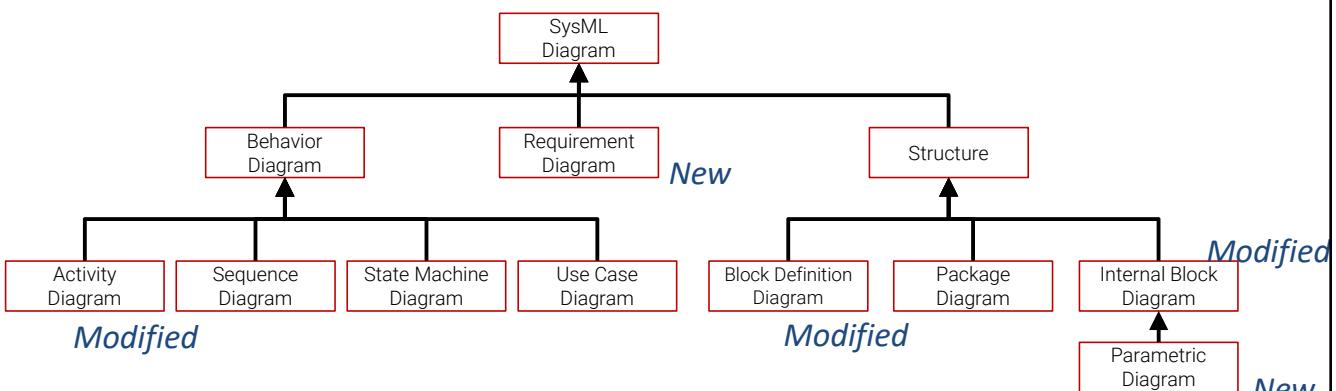
FH W-S  
University of Applied Sciences  
Würzburg-Schweinfurt

10





## SysML Diagram Taxonomy

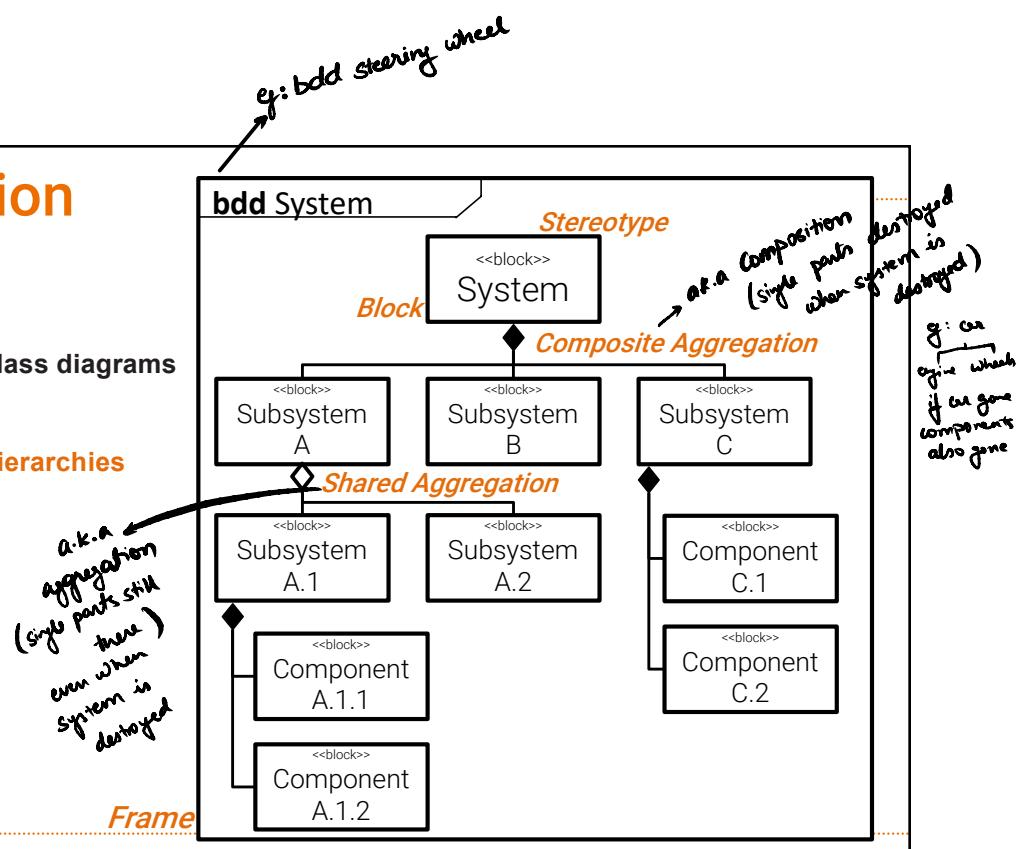


[OMG Systems Modeling Language v1.6, 2019]

# Block Definition Diagram

BDDs are based on UML class diagrams

BDDs are used to model hierarchies



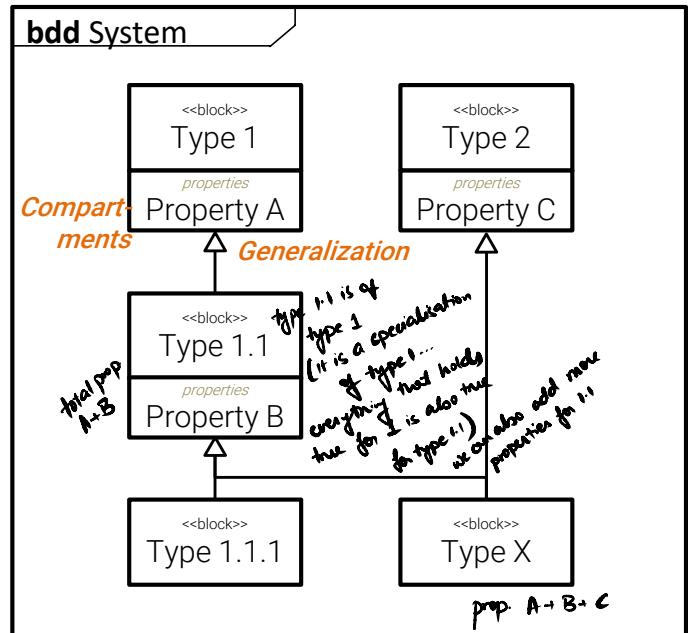
# Block Definition Diagram

BDDs are based on UML class diagrams

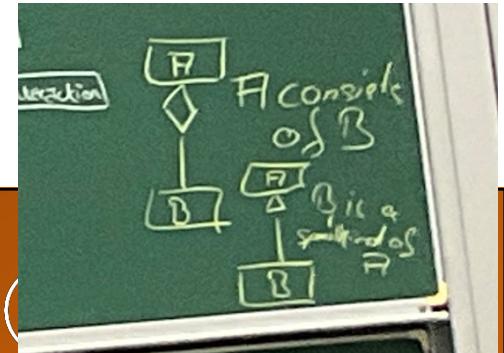
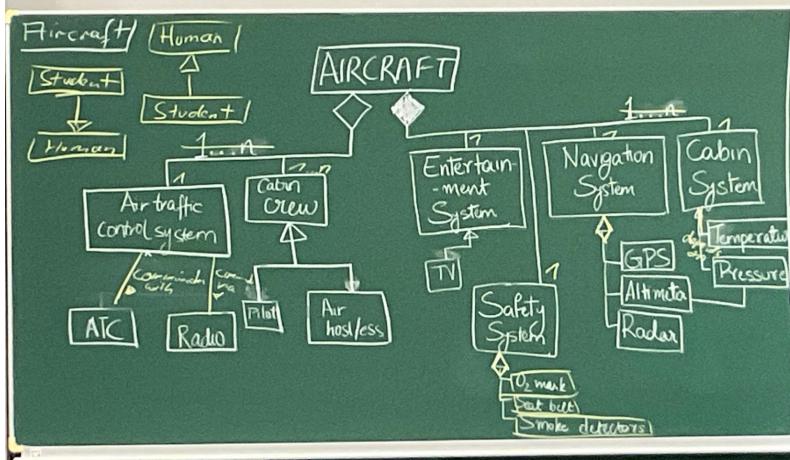
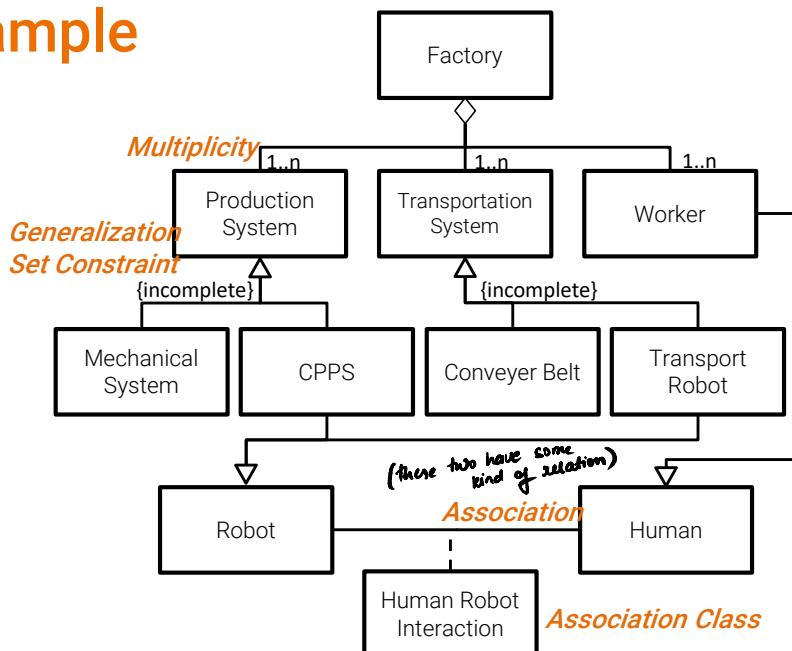
BDDs are used to model hierarchies

BDDs are used to model generalization/  
specialization relations

*multiple inheritance*

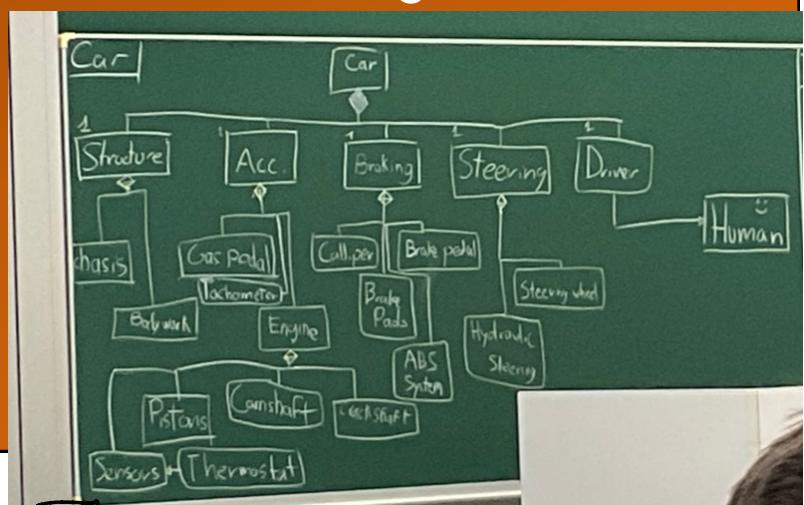


## Example

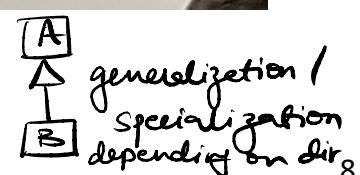


## Create SysML Block Definition Diagrams for:

- An Aircrafts
- A Car
- A Smart Factory
- Autonomous Driving



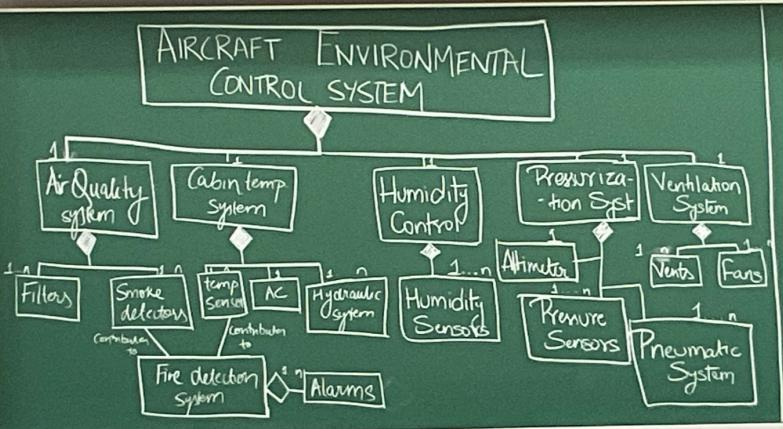
all compositions  
are aggregations  
but all agg. is  
not composition



# Scope of Development

From the point of an original equipment manufacturer a car might be the system under development. However, it is uncommon to refer to a car as the system. Typically, the system under development is a system embedded in the car.

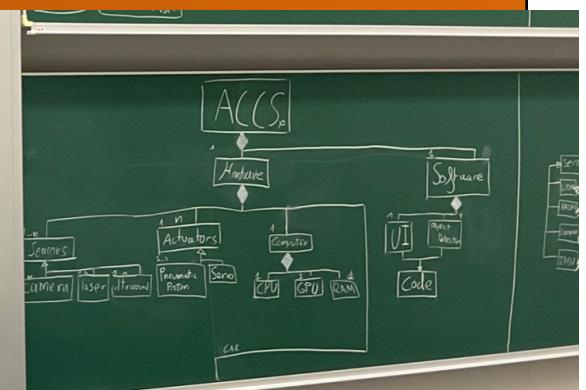
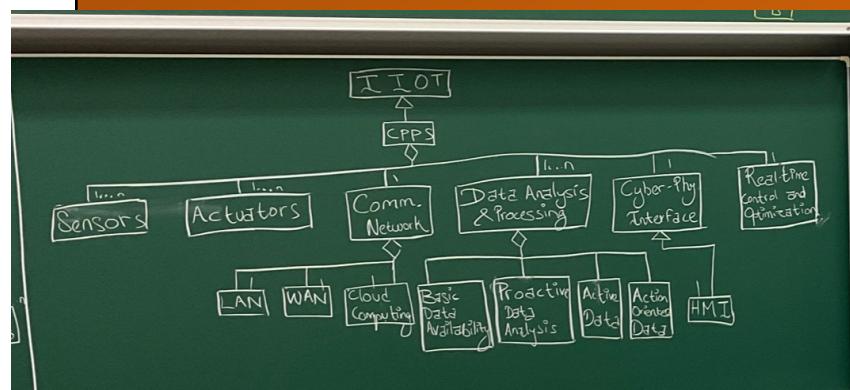
An aircraft, a car, a smart factory are thus super systems.  
For autonomous driving a platoon is a system of systems.



Exercise

## Create SysML Block Definition Diagrams for:

- An embedded system of an Aircraft
- An embedded system of a Car
- A cyber-physical production system within a Smart Factory
- The transport system of a smart factory as a System of Systems





Group Discussion

**What do we notice when comparing the model of the super system with the model of the embedded system?**

## Context

# Context is Important

Separation of **Context** and **System** allows to differentiate between the scope of development (i.e. **what can be changed**) and the **constraining** environment

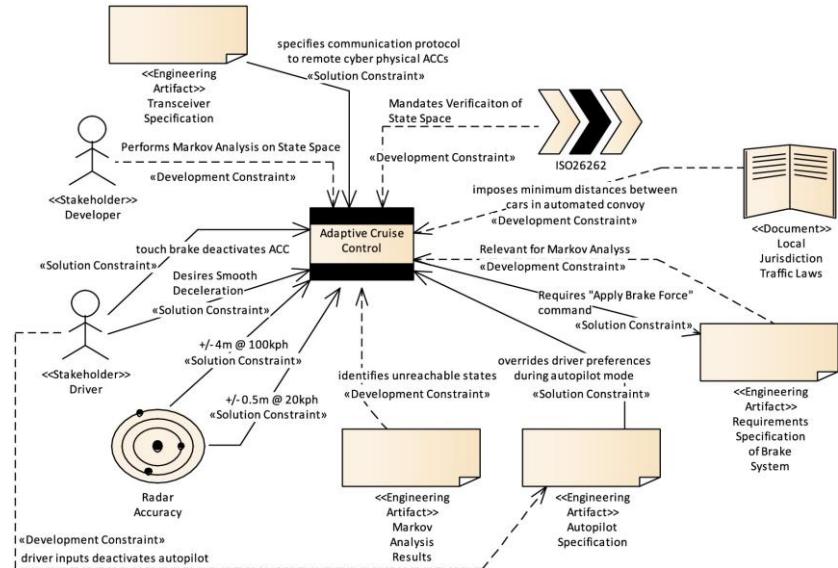
- Context is main source of requirements and rationales
- Context is source of safety and security risks/threats

**Context changes** can severely affect the system

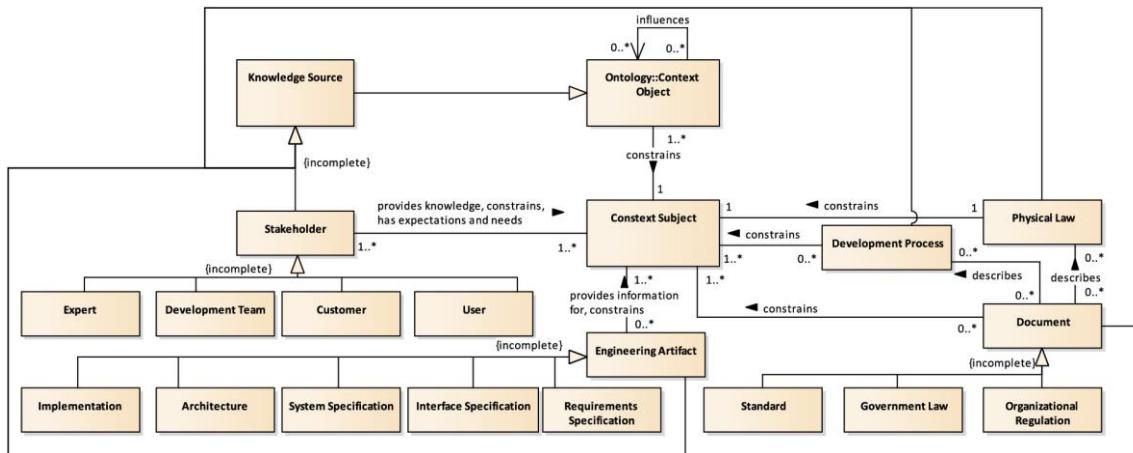
- Functionality
- Correctness
- Safety
- Security

# Context of Knowledge

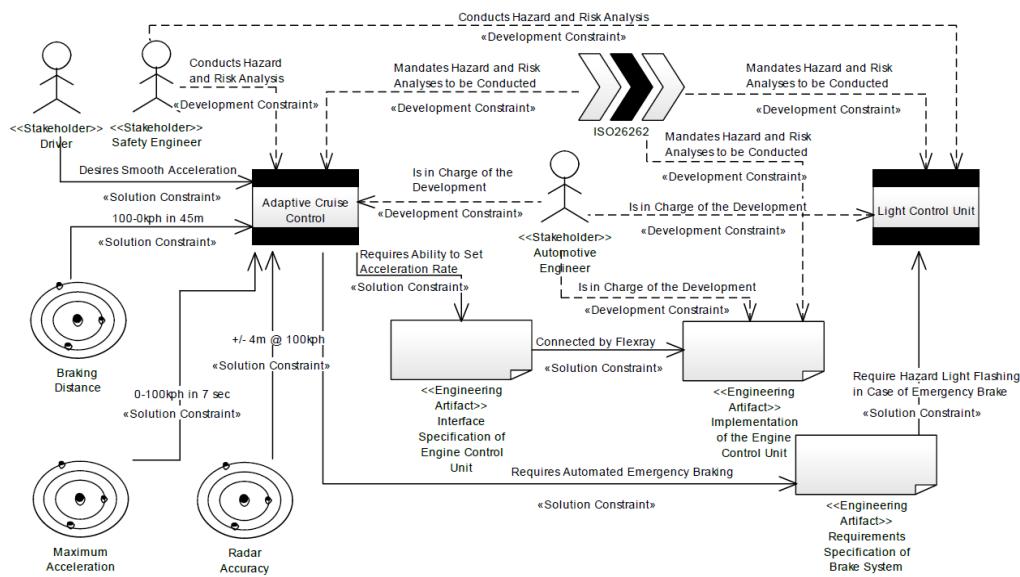
The context of knowledge defines the major requirements sources and important knowledge bearers for the development of a system.



# Context of Knowledge Meta Model



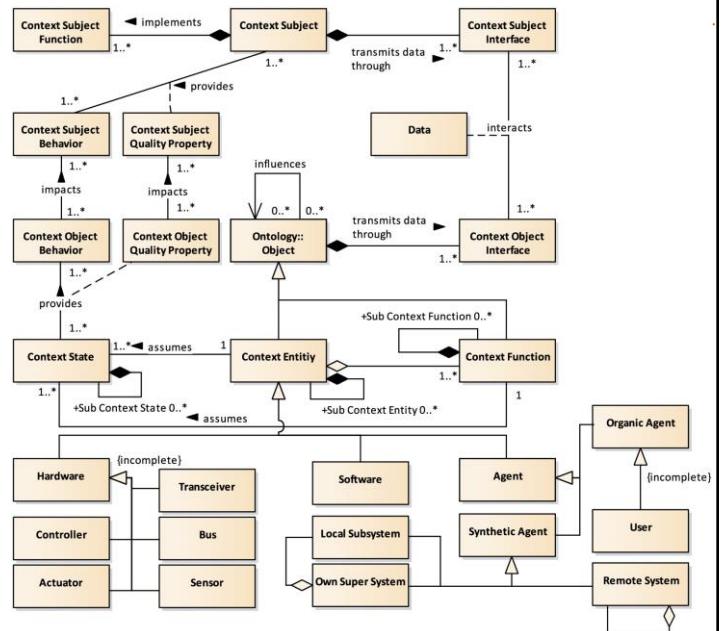
# Combining different Contexts



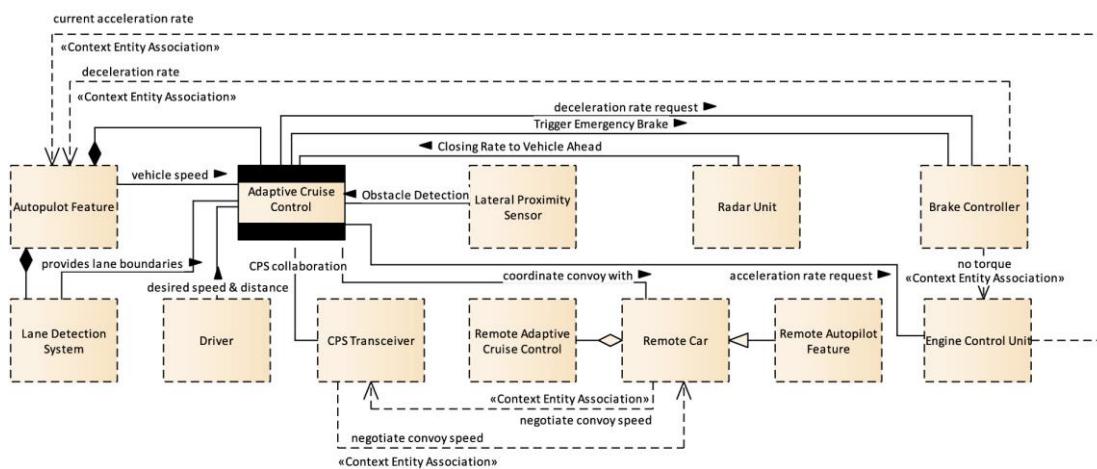
# Operational Context

The operational context focusses on context objects that interact with the system during runtime.

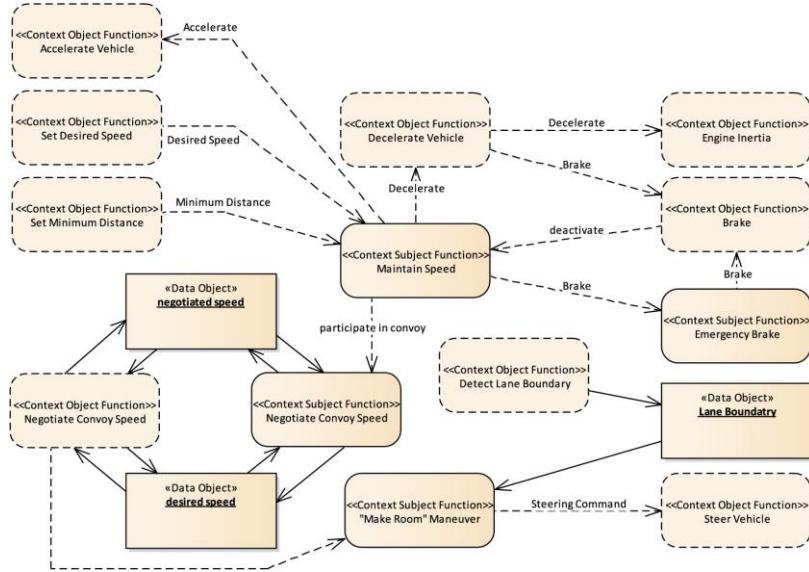
Thus, consideration of the operational context is important for defining interfaces and system behavior in accordance with its environment



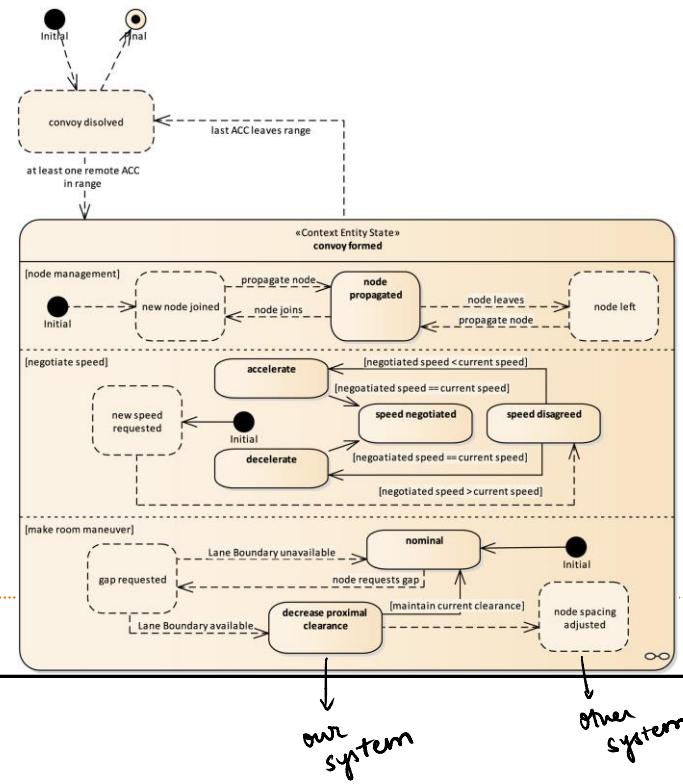
# Structural Operational Context



# Functional Operational Context



# Behavioral Operational Context



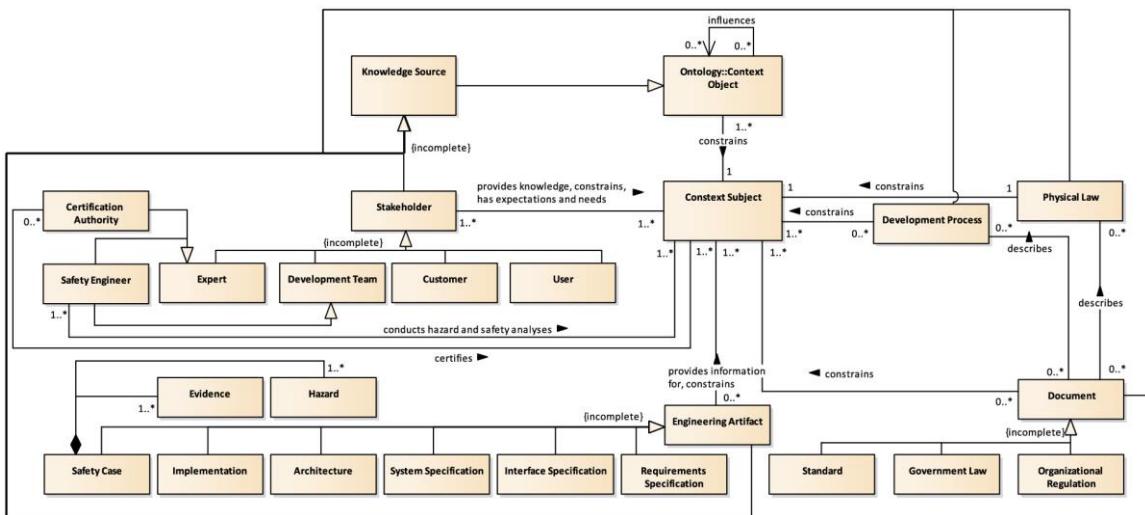


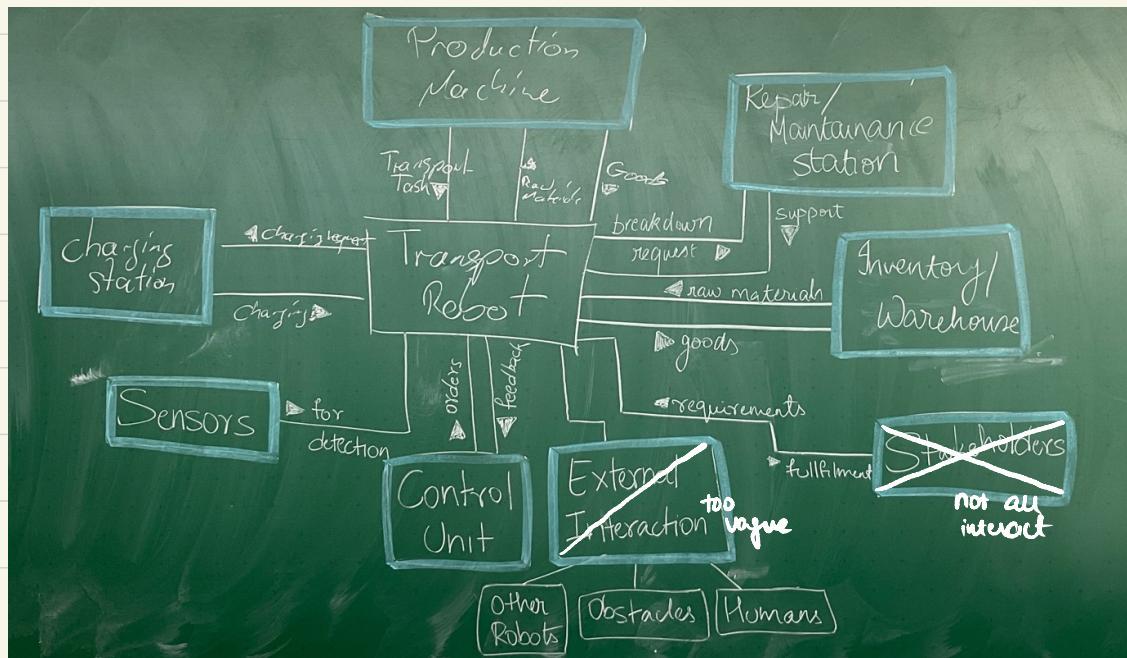
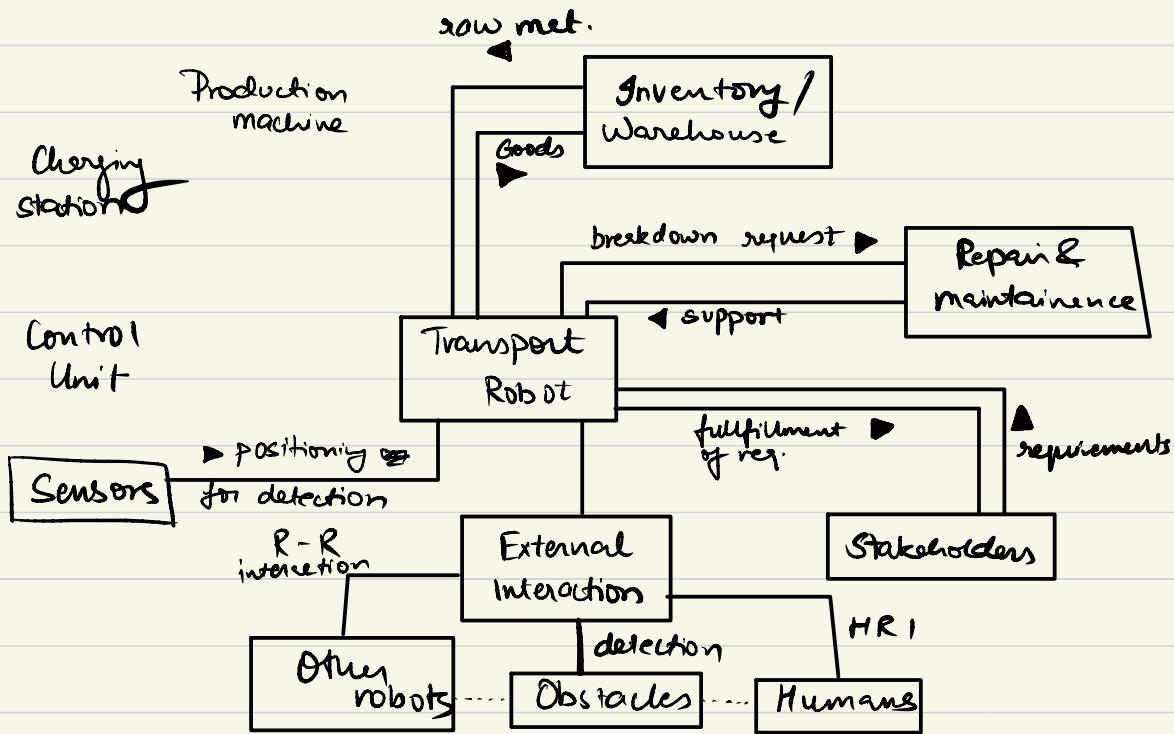
## Exercise

# Define the Context of

- A Cobot
- A Transport Robot

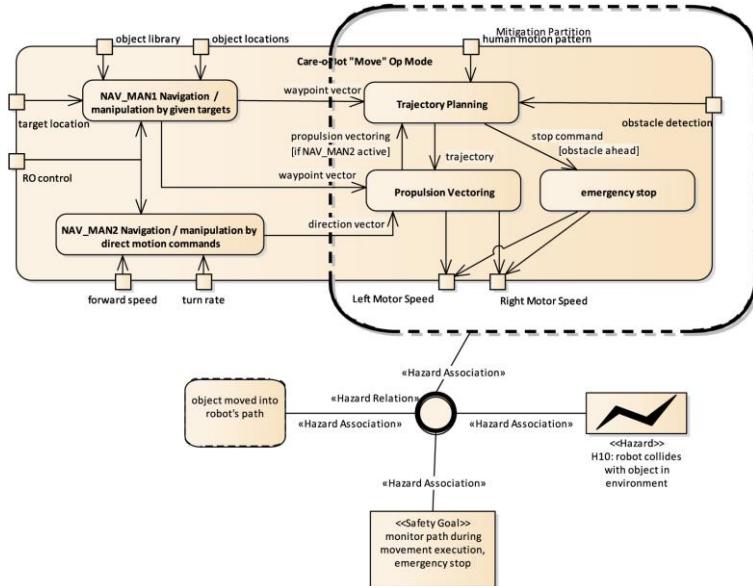
## Safety Analyses







# Hazard Modelling



Exercise

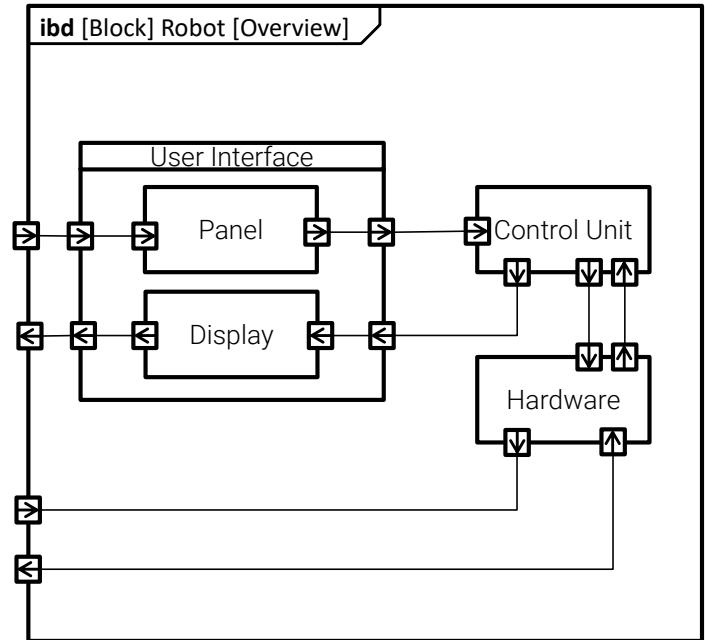
**Identify Safety Hazards for**  
**- A Cobot**  
**- A Transport Robot**

# Modelling Structure with SysML Internal Block Diagrams

## Internal Block Diagram

IBDs define the internal structure

IBDs define interfaces



## What is an IBD?

**Internal Block Diagram (ibd):** An *Internal Block Diagram* is a static structural diagram owned by a particular Block that shows its encapsulated structural contents: Parts, Properties, Connectors, Ports, and Interfaces. Stated otherwise, an IBD is a "white-box" perspective of an encapsulated ("black-box") Block.

- Blocks can be recursively decomposed ("nested") into Parts by alternating between Block Definition Diagram (BDD) *definitions* and Internal Block Diagram (IBD) usages (See *Usage Notes* below.)
- Behaviors can either be encapsulated by Blocks (e.g., Operations, Signals, and State Machines) or Allocated (via «allocate» Dependency) to Blocks (e.g., Activities/Actions) directly or indirectly (via Interfaces).
- [...]

From: <https://sysml.org/sysml-faq/what-is-internal-block-diagram.html>

## Purpose of IBDs

The purpose of Internal Block Diagrams (IBDs) is to show the encapsulated structural contents (Parts, Properties, Connectors, Ports, Interfaces) of Blocks so that they can be recursively decomposed and "wired" using Interface Based Design techniques. [...]

From: <https://sysml.org/sysml-faq/what-is-internal-block-diagram.html>

## BDD vs IBD

### BDD Block *Definition* vs. IBD Block *Usage* Dichotomy

BDDs and IBDs complement each other (cf. black-box vs. white-box) and support recursive structural decomposition techniques during System Analysis & Design.

- A BDD *defines* a Block's Properties, including its Part Properties (strongly owned Parts) and Reference Properties (shared Parts)
- IBD specifies Part Properties and Reference Properties *usages* or roles in the structural context of the Block that encapsulates them. Stated otherwise, Part Properties and Reference Properties in an IBD can have a different usages or roles depending upon how they are *realized* ("wired") in the IBD.
- [...]

From: <https://sysml.org/sysml-faq/what-is-internal-block-diagram.html>



**Exercise**

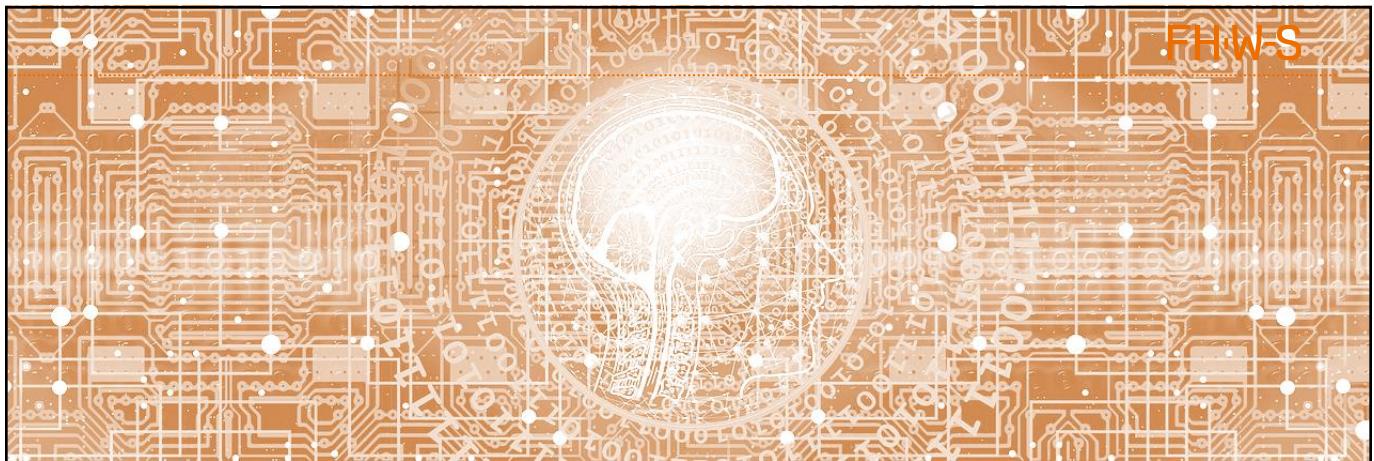
**Model BDDs and IBDs for describing**  
**- A Cobot**  
**- A Transport Robot**

## Questions for Self-Assessment

- What is a Block Definition Diagram?
- What is an Internal Block Diagram?
- How do BDDs and IBDs complement each other?
- What are important system layers in the development of embedded systems?
- What influences the definition of components?
- Why is context important?
- How can context be used to identify safety hazards?

## Literature

- |                           |  |
|---------------------------|--|
| [Daun et al. 2016]        | M. Daun, B. Tenbergen, J. Brings, T. Weyer: SPES XT Context Modeling Framework. In: K. Pohl, M. Broy, H. Daembkes, H. Hönniger (eds.) Advanced Model-Based Engineering of Embedded Systems, Springer, 2016, pp. 43-57. |
| [Daun & Tenbergen 2022]   | M. Daun, B. Tenbergen: Context modelling for cyber-physical systems. In: J. Softw. Evol. Proc., Wiley, 2022.   |
| [Friedenthal et al. 2014] | Sanford Friedenthal, Alan Moore & Rick Steiner: A Practical Guide to SysML – The Systems Modeling Language. Morgan Kaufmann OMG Press, 2014.   |
| [OMG SysML]               | OMG System Modeling Language. Version 1.6, Object Management Group, 2019.  |
| [OMG UML]                 | OMG Unified Modeling Language. Version 2.5.1, Object Management Group, 2017.   |



# Embedded Systems and Field Buses

Prof. Dr. Marian Daun

**FHWS** University of Applied Sciences  
Würzburg-Schweinfurt

## Agenda

- Fundamentals
- Structure of Embedded Systems
- Behavior of Embedded Systems
- Design of Embedded Systems
- Communication
- Real-time
- Collaborative Embedded Systems

**FHWS** University of Applied Sciences  
Würzburg-Schweinfurt

# Behavior of Embedded Systems

Part A: Petri Nets



Group Discussion

## Why is Structure not enough?

# Formal Languages

Behavior is typically described using formal languages.

The use of formal languages allows (among others)

- Formal **Proof of Concept**
- **Simulation**
- Analyses (e.g. **Deadlock** and **Lifelock** detection, finding an **Equilibrium**)
- **Verification**
- Test Case Generation
- ...

# Automata Theory

# Finite State Machines

Automata typically have

- A **formal** representation
- A **graphical** representation

Automata foremost consist of

- **States**
- **Transitions**

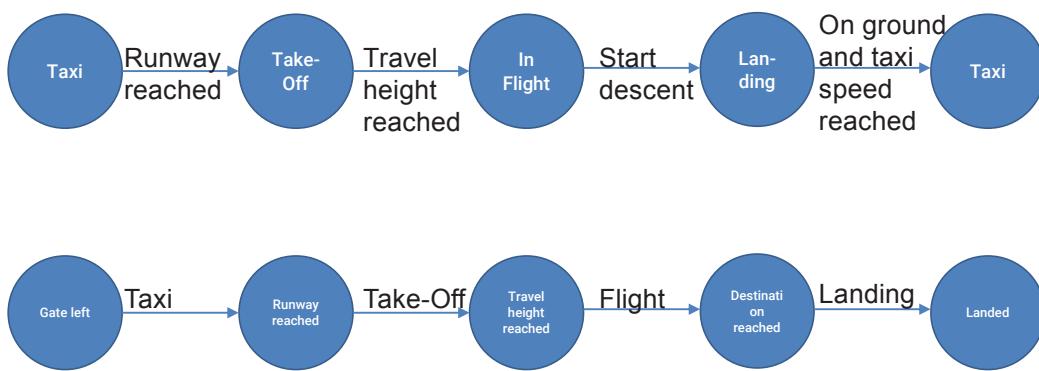
Further concepts include:

- Start/initial states
- Final/accepting states
- Events and Conditions

Depending on the language the focus is either on

- the **States**
- the **Transitions**
- or **Both**

# State vs Transition





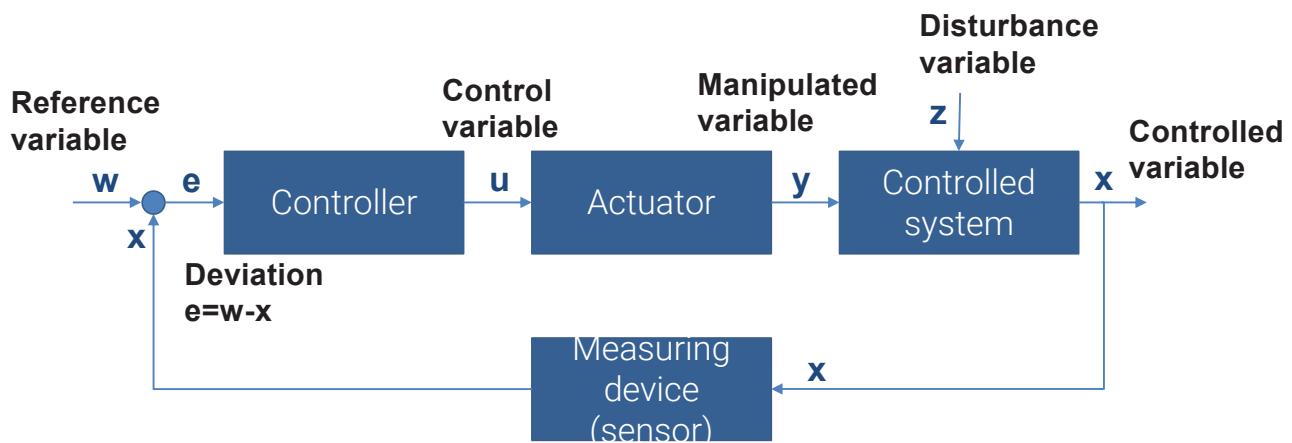
Group Discussion

## What is an Alphabet?

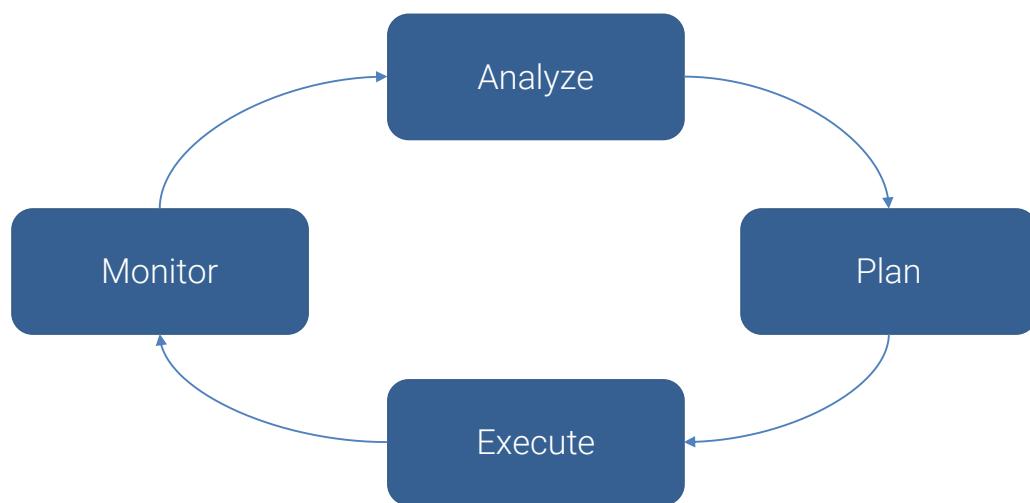
## What are basic Operators?

## Remember Control- and MAPE-Loops

## Control Loop



## MAPE-Loop

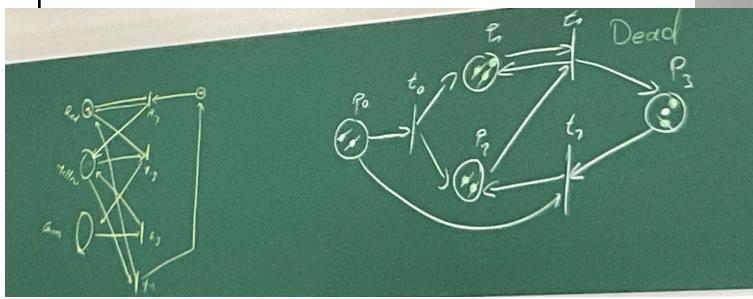
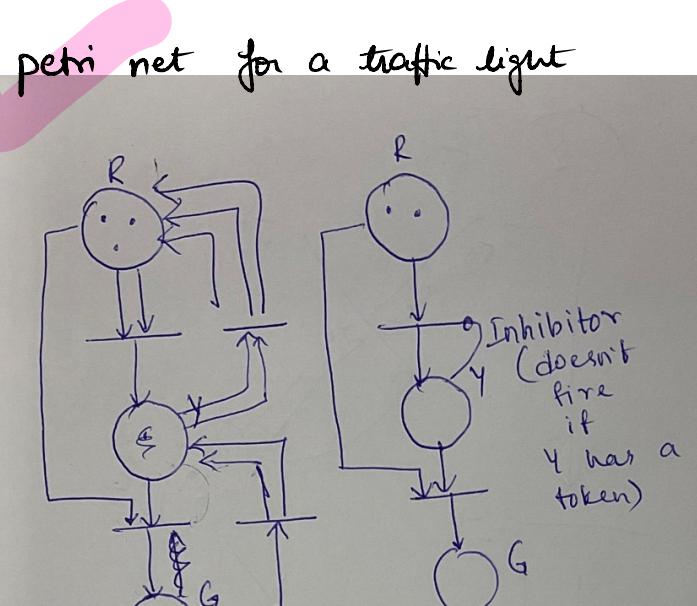


# Petri Nets

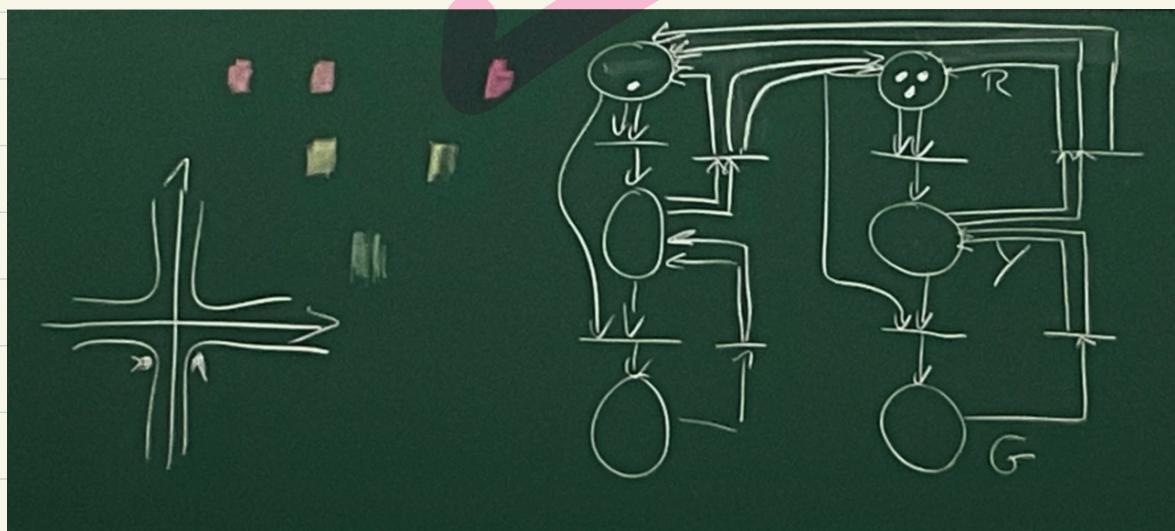
## Online-Materials

Introduction videos to Petri Nets can be found at:

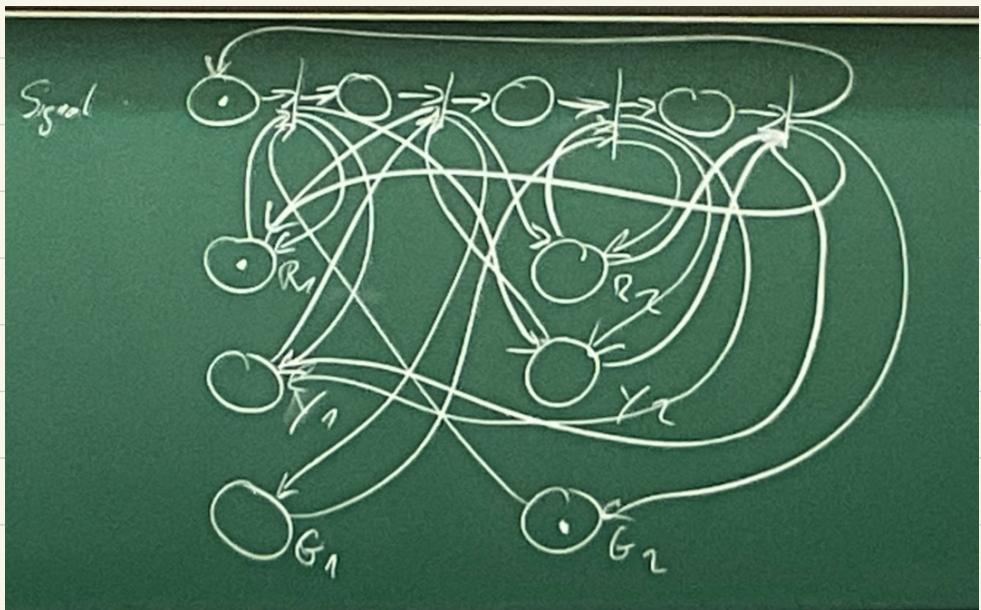
- <https://youtu.be/GCsVxWh995o>
- <https://youtu.be/WGSAi9-QUwk>
- <https://youtu.be/1IPOIE0PvQY>

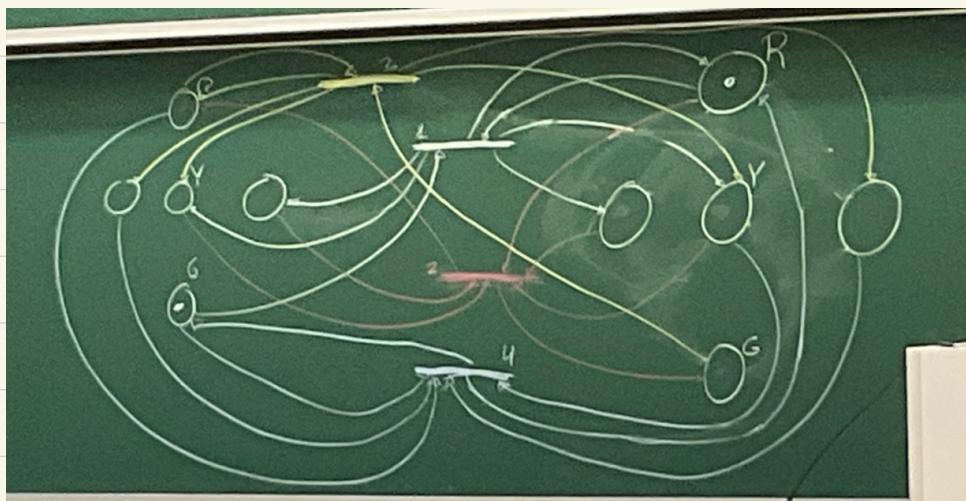


2 traffic lights

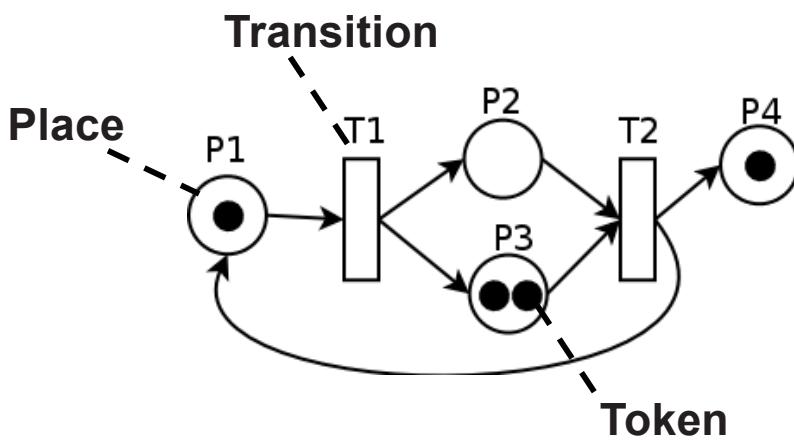


synchronized ↴



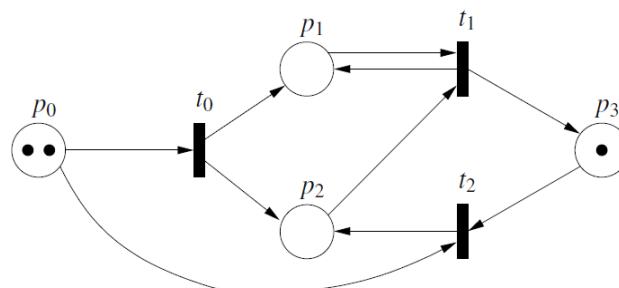


## Elements



## Petri nets

Is transition  $t_2$  of the petri net shown below enabled? (Capacity of places is  $\infty$ )  
 What is the marking after  $t_2$  fires?





## Exercise

# Model a Traffic Light using Petri Nets

## Analysis

Terms:

A transition is **dead**, if it is not enabled in any marking.

A transition  $t$  is potentially **fireable**, when there is at least one marking reachable, which enables  $t$ .

A petri net is **dead**, if no transition is able to fire in a given marking.

A petri net is **quasi alive**, if not dead under any future marking.

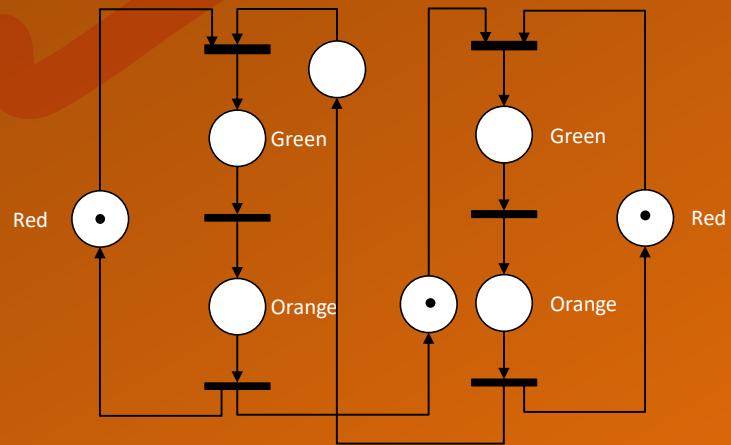
A petri net is **alive**, when all transitions are potentially fireable for all future markings.



### Exercise

**Quasi alive  
or alive?**

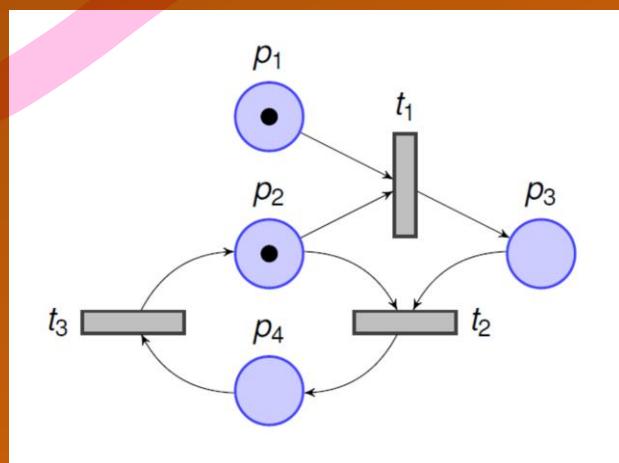
alive  
all are potentially  
fireable

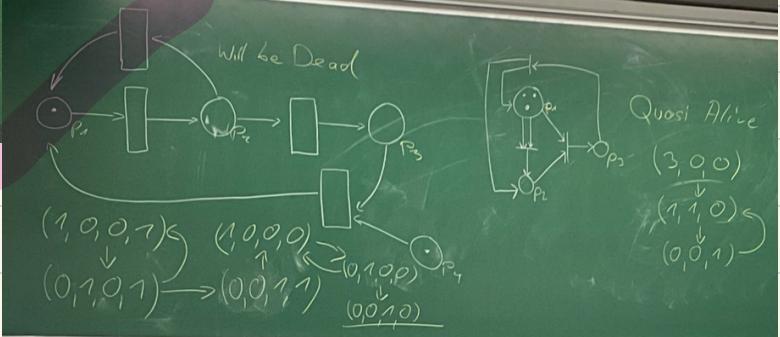
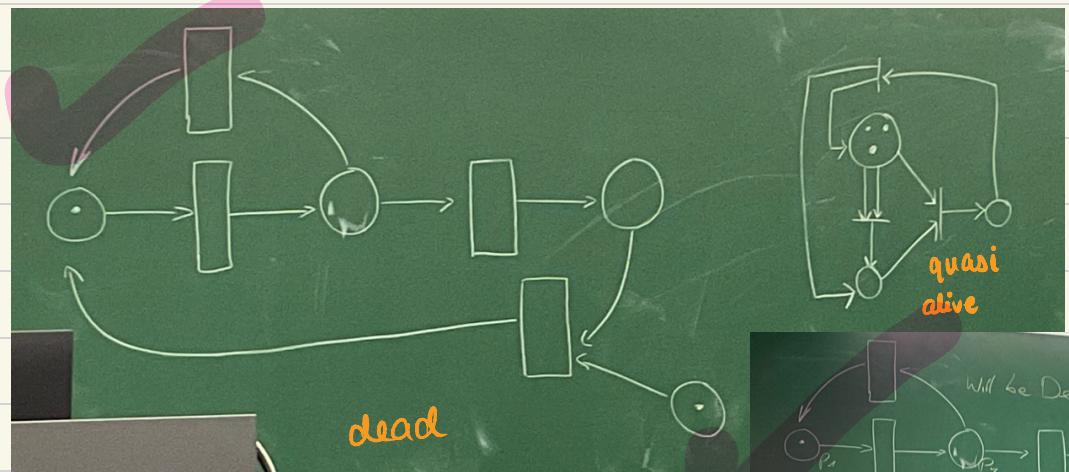
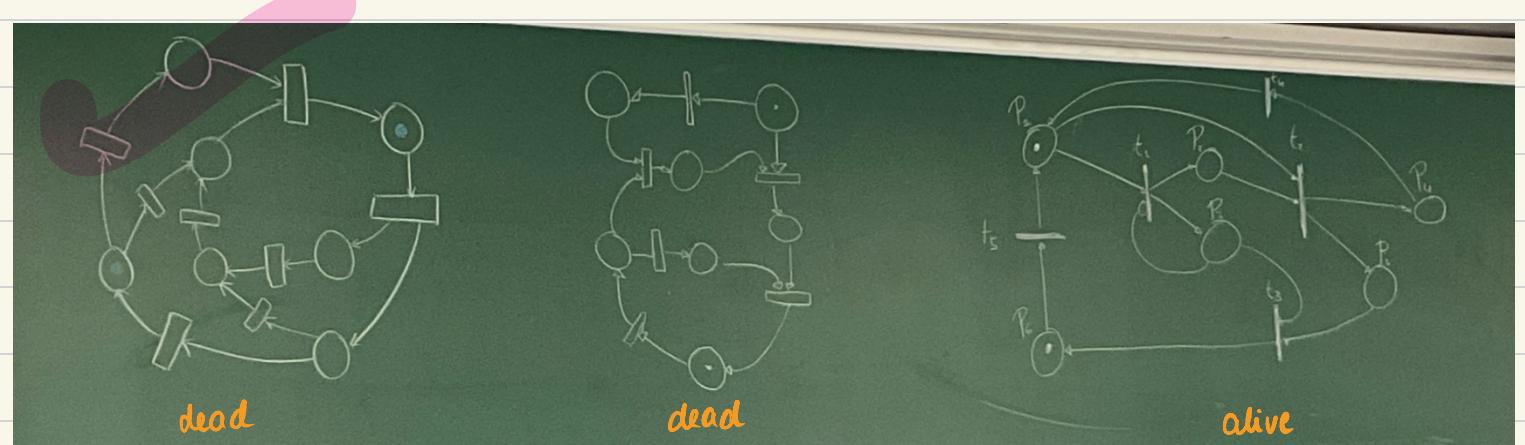
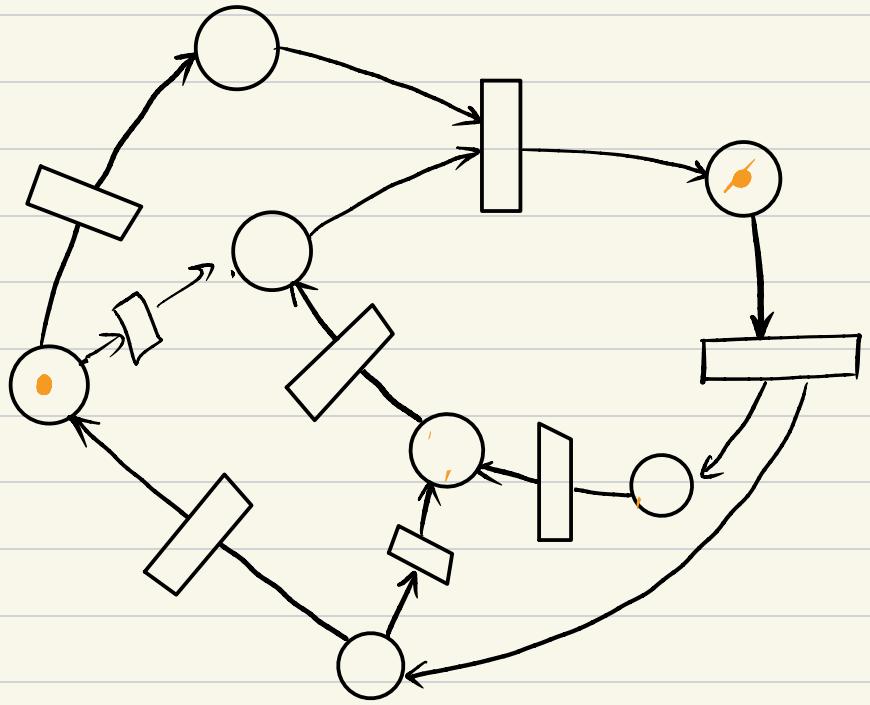


### Exercise

**Quasi alive  
or alive?**

dead  
After  $P_1$  is fired it  
can't be fired again.

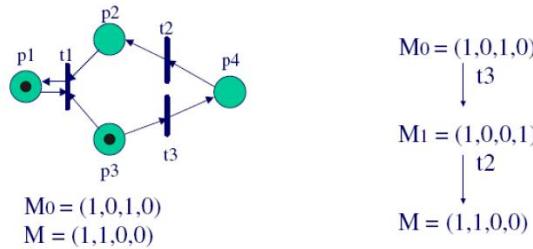




## Reachability Graph

Firing sequence for a petri net  $G$  and initial marking  $M_0$  is a sequence of transitions  $t_0, \dots, t_n$ , such that  $M_0 \xrightarrow{G, t_0} M_1 \xrightarrow{G, t_1} \dots \xrightarrow{G, t_n} M_n$

**Reachability graph:** The reachability graph of  $G$  is the transition relation  $\xrightarrow{G}$  restricted to its reachable markings  $R(G)$ . It is the state space of the net.



Nassar, K., & Casavant, A. (2008). Analysis of timed Petri nets for reachability in construction applications. *Journal of Civil Engineering and Management*, 14(3), 189-198.

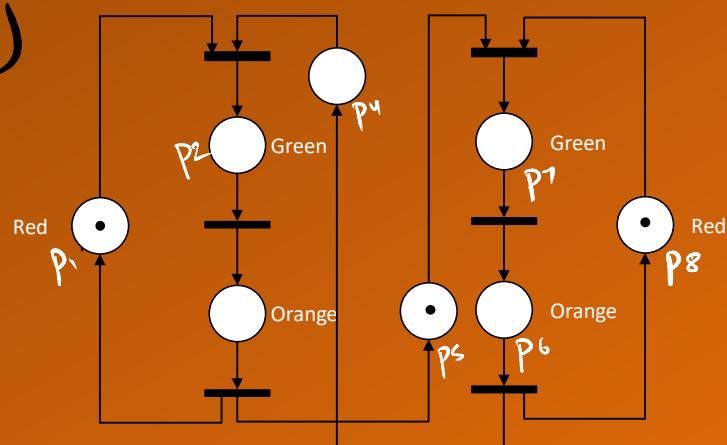
$M_0 = (1\ 0\ 0\ 0\ 1\ 0\ 0\ 1)$   
 $M_1 = (1\ 0\ 0\ 0\ 0\ 0\ 1\ 0)$   
 $M_2 = (1\ 0\ 0\ 0\ 0\ 1\ 0\ 0)$   
 $M_3 = (1\ 0\ 0\ 1\ 0\ 0\ 0\ 0)$

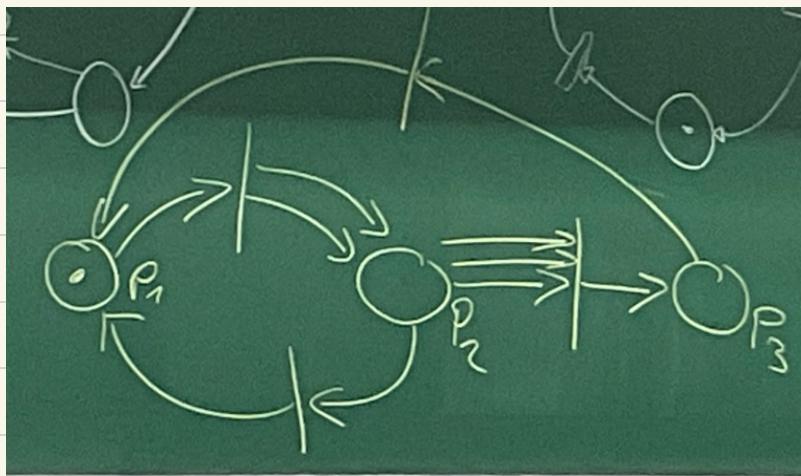
Draw the Reachability

$(1, 0, 0, 0, 1, 0, 0, 1) \leftarrow$   
 $\downarrow$   
 $(1, 0, 0, 0, 0, 1, 0, 0)$   
 $\downarrow$   
 $(1, 0, 0, 0, 0, 0, 1, 0)$   
 $\downarrow$   
 $(1, 0, 0, 1, 0, 0, 0, 1)$   
 $\downarrow$   
 $(0, 1, 0, 0, 0, 0, 0, 1)$   
 $\downarrow$   
 $(0, 0, 1, 0, 0, 0, 0, 1)$



Exercise





$$M_0 = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$$

$$M_1 = \begin{pmatrix} 0 & 2 & 0 \end{pmatrix}$$

$$M_2 = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix}$$

$$N_3 = \begin{pmatrix} 0 & 3 & 0 \end{pmatrix}$$

✓

$$M_4 = \begin{pmatrix} 1 & 2 & 0 \end{pmatrix}$$

$$M_5 = \begin{pmatrix} 0 & 4 & 0 \end{pmatrix}$$

$$M_6 = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}$$

$$M_7 = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$$

$$M_6 = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}$$

$$M_7 = \begin{pmatrix} 0 & 2 & 1 \end{pmatrix}$$

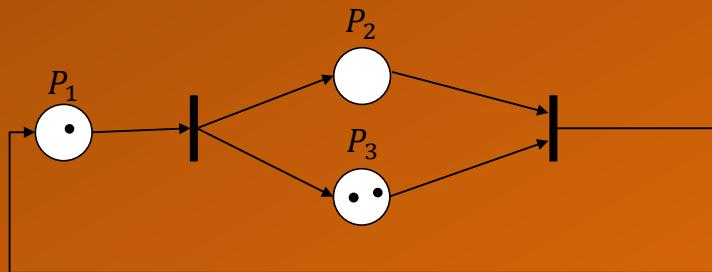
$$M_8 = \begin{pmatrix} 1 & 2 & 0 \end{pmatrix}$$

## Draw the Reachability Graph

$$\begin{aligned}M_0 &= \begin{pmatrix} 1 & 0 & 2 \end{pmatrix} \\M_1 &= \begin{pmatrix} 0 & 1 & 3 \end{pmatrix} \\M_2 &= \begin{pmatrix} 1 & 0 & 2 \end{pmatrix}\end{aligned}$$



**Exercise**



**Group Discussion**

**Why are Petri Nets commonly suggested for analysing embedded systems?**

**Where are Limitations of Petri Nets?**



## Exercise

For home:  
**Google the Dining Philosophers Problem  
and Model it with Petri Nets**

## Questions for Self-Assessment

- What are the basic concepts of automata theory?
- What is a petri net?
- When does a petri net fire?
- Are petri nets deterministic?
- When is a petri net dead, alive, quasi alive?
- What is a reachability graph?
- What are shortcomings of petri nets?
- Why are petri nets often suggested for modelling embedded systems?

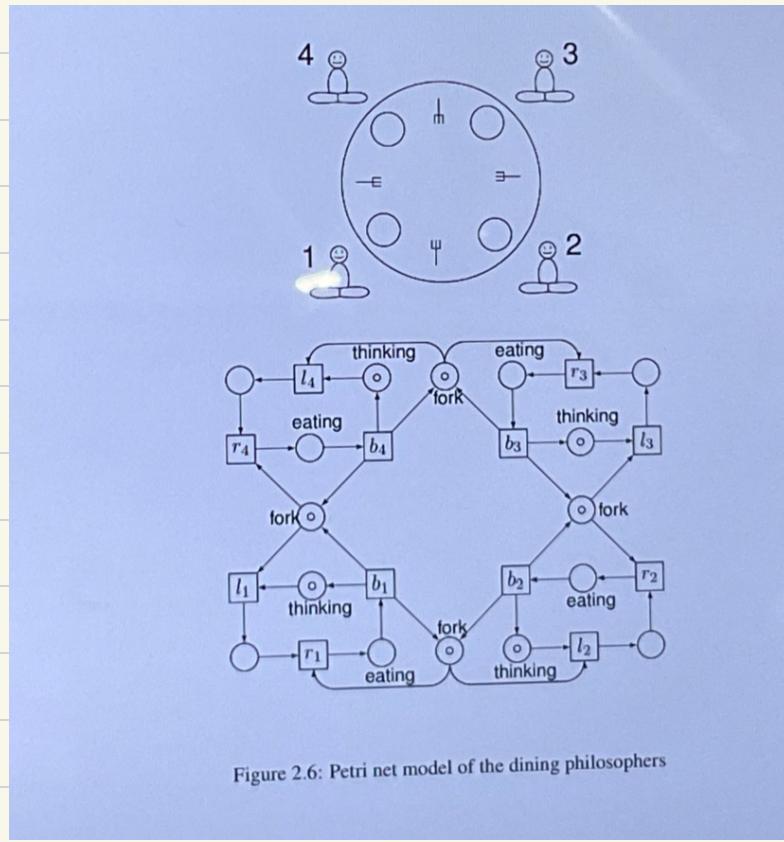
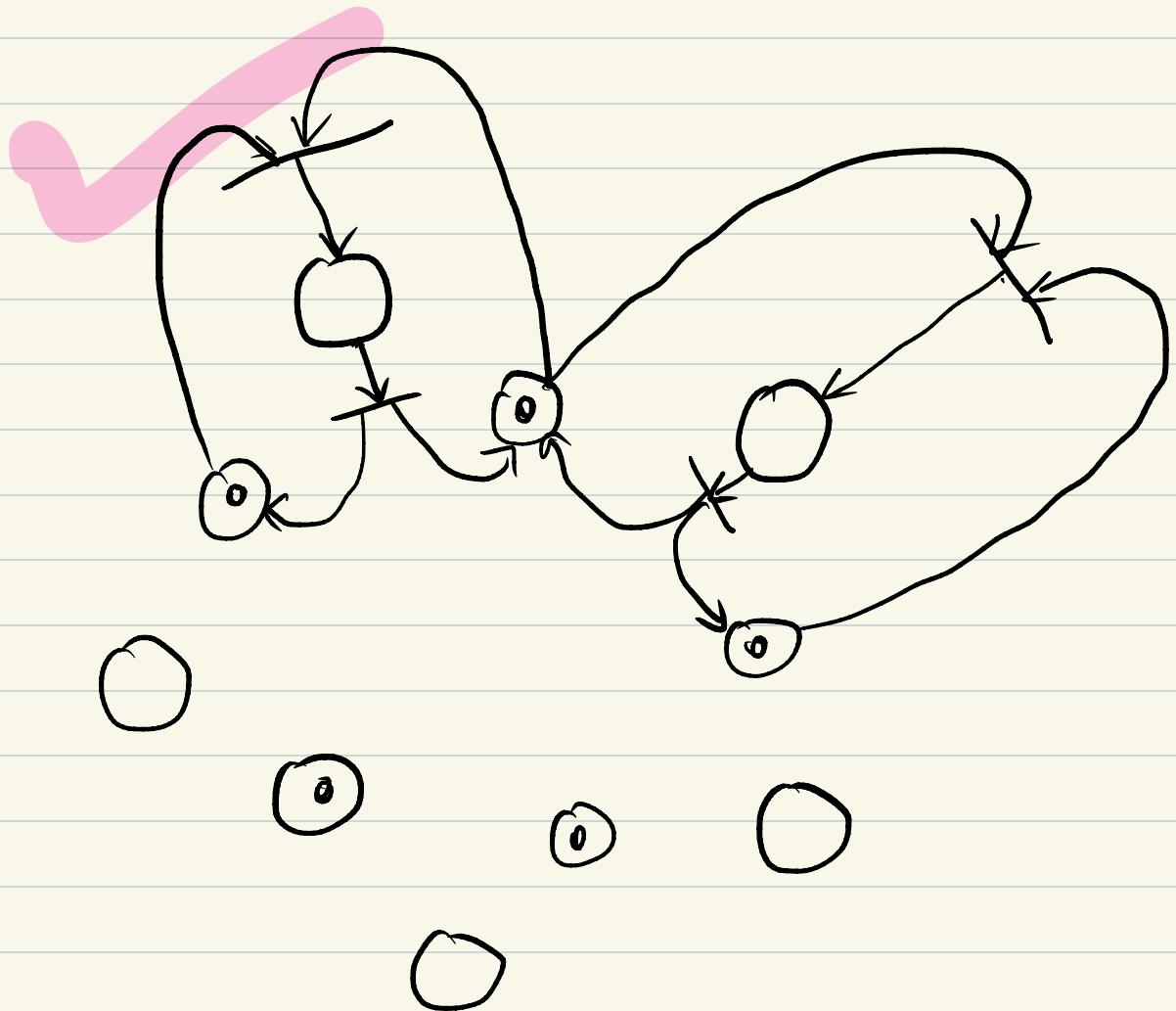
## Literature

[Hopcroft et al. 2006]

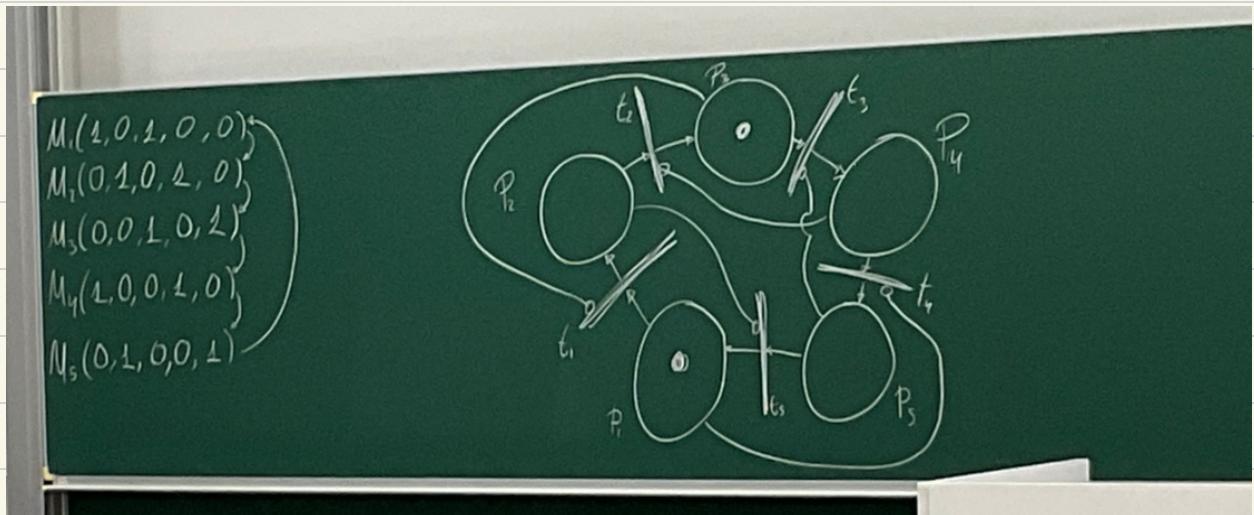
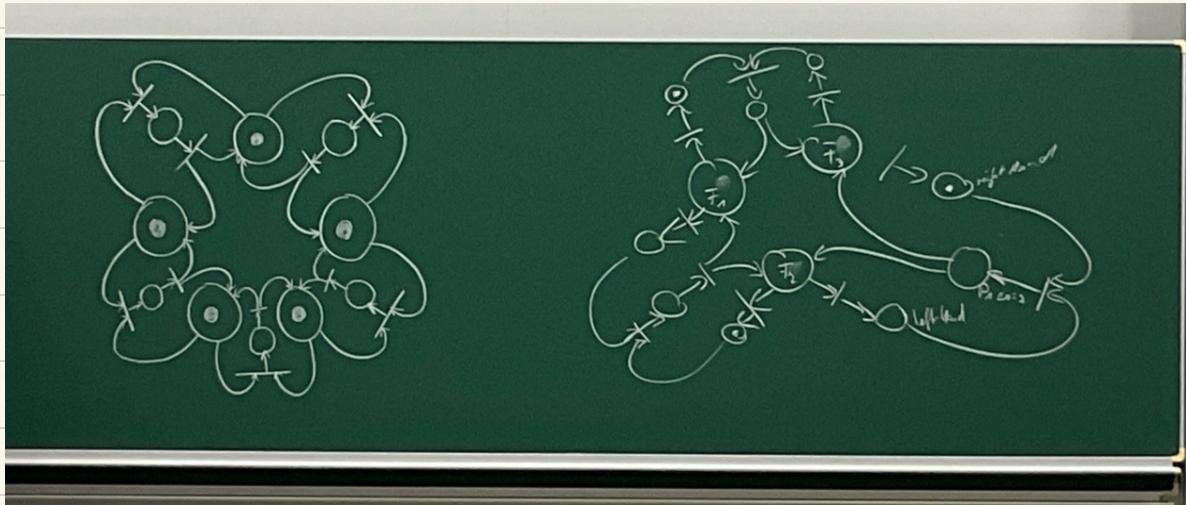
Introduction to Automata Theory, Languages, and Computation. 3<sup>rd</sup> Edition, Pearson Education, 2006.

[Reisig 1991]

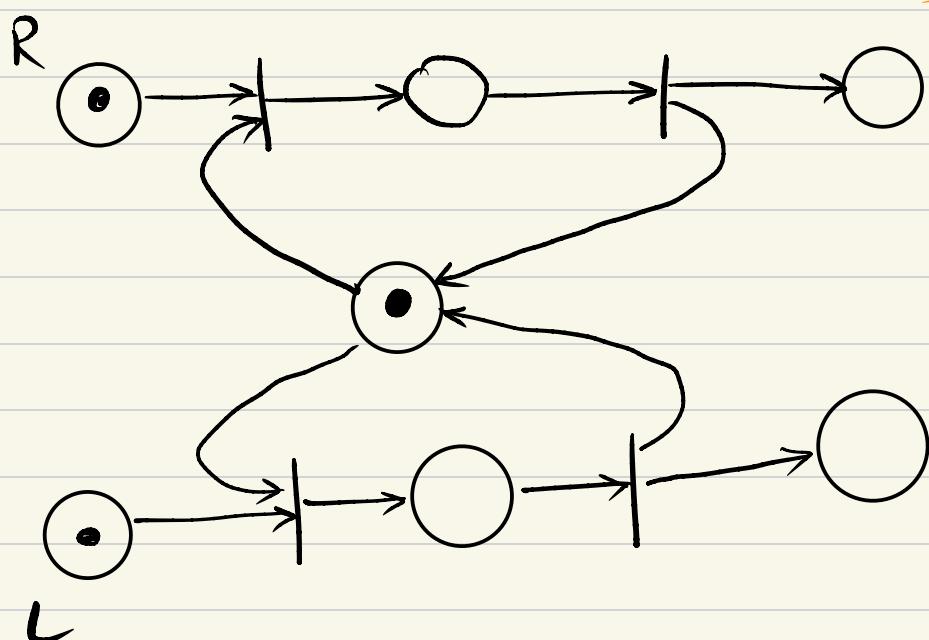
Reisig, W.: Petri Nets and Algebraic Specifications. In: Theoretical Computer Science, 80(1), 1991, pp. 1-34.



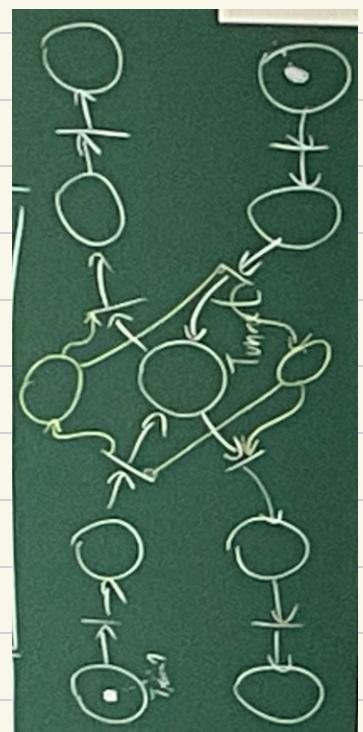
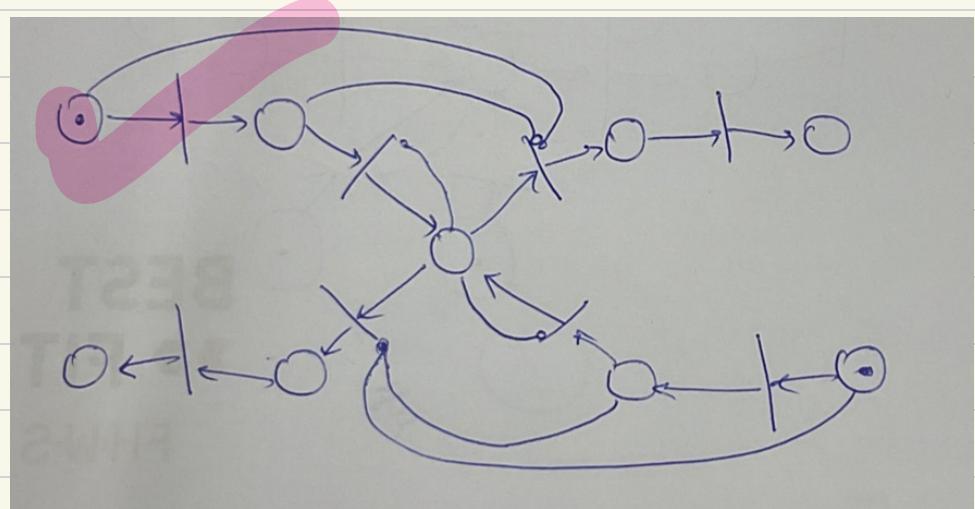
Q) Model Petri nets for Dining Philosophers Problem



Q Petri nets for

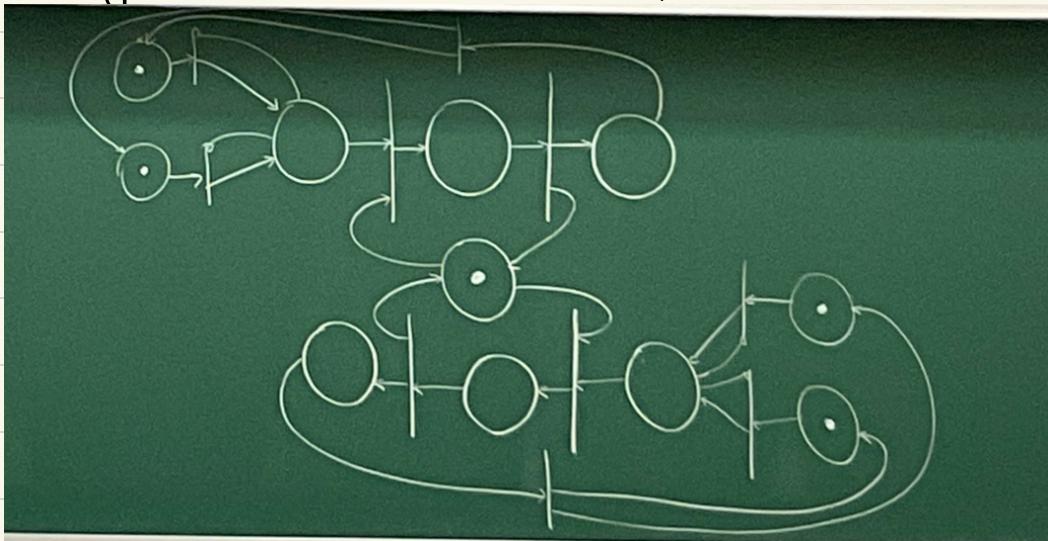


two trains  
enter a tunnel  
and only  
one can  
pass



Q Two trains meet at a junction and have an option to go either left or right.

(wry answer probably)

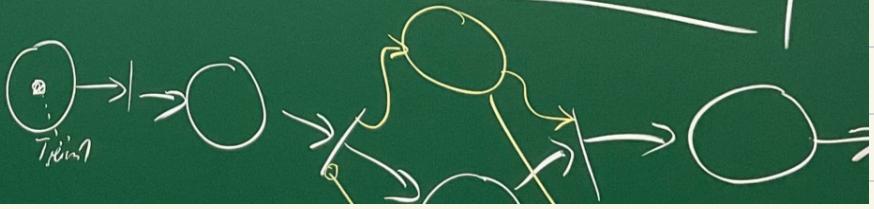


You specify a water fountain. The water fountain shall turn on if someone is in its vicinity and it is between 7am and 1am. Lights shall turn on if movement is detected and it is after dark (before sunrise).

A two track train road shares a single track tunnel. Make sure that two trains travelling in opposite directions can safely pass.

A system of slaves in memory with an independent clock. It consists of a master and two slaves. One of them does their job they can go again. Master device is clock dependent.

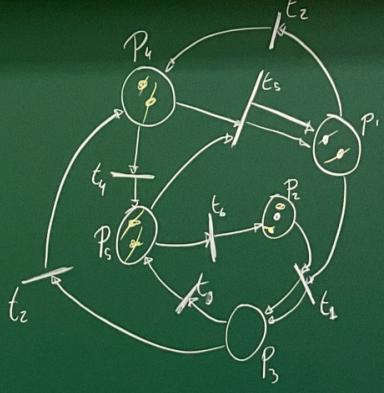
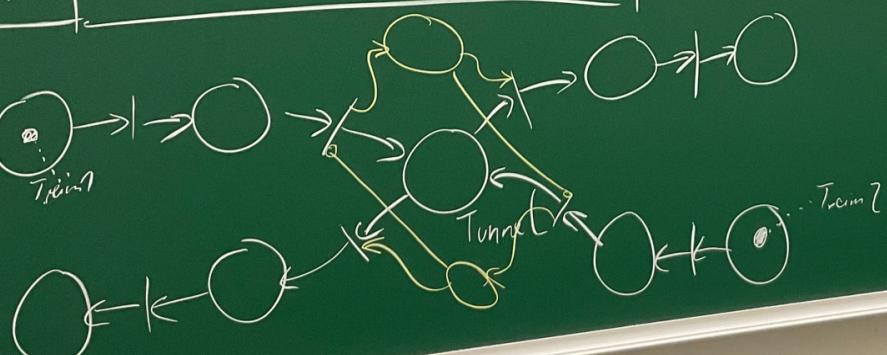
Slave device is choosing the slave at random. Slave device is not clock dependent.



- fountain  
shall turn on if  
it is after dark  
(before sunrise).

A two track train road shares a single track tunnel. Make sure that two trains travelling in opposite directions can safely pass.

slaves  
independent



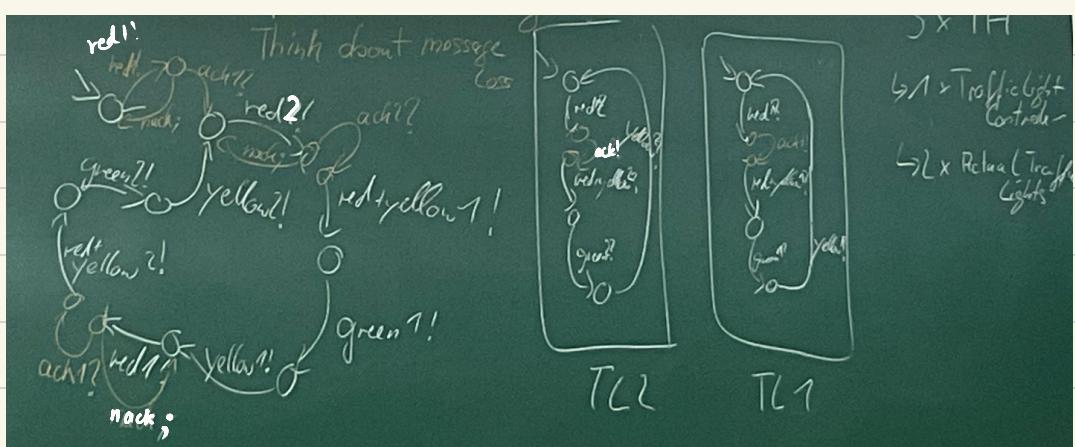
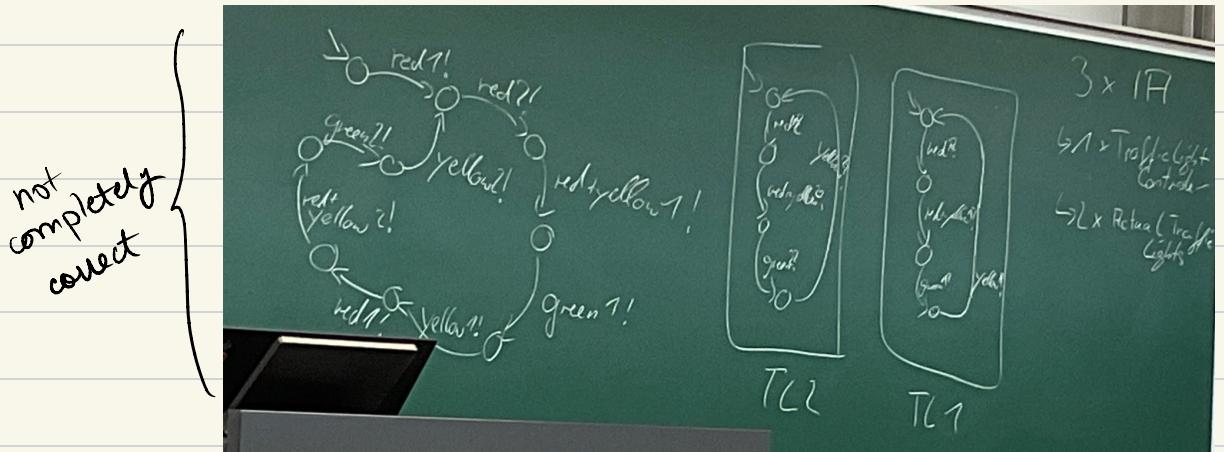
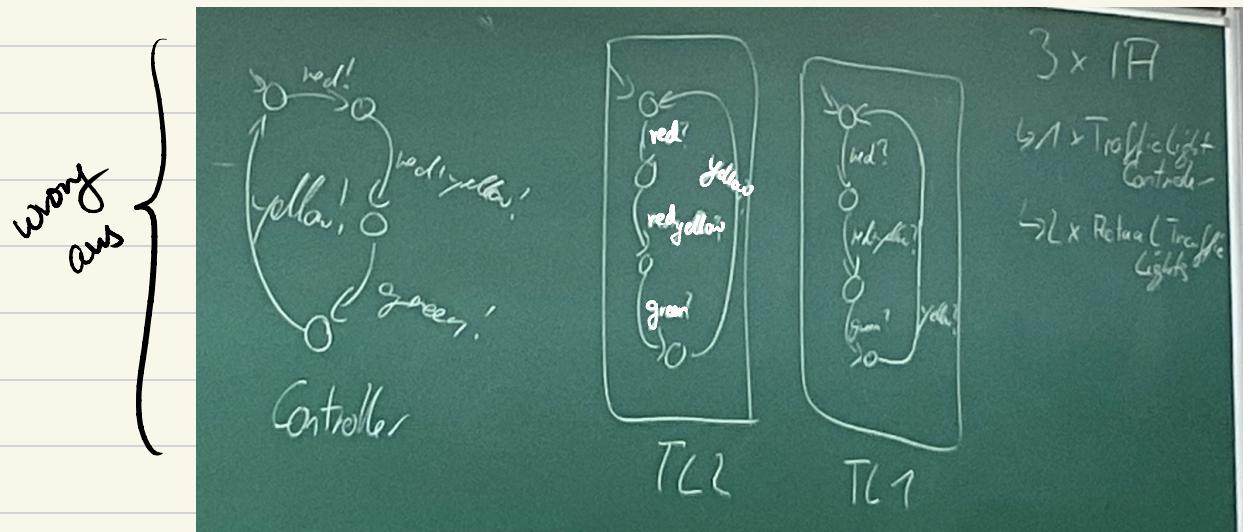
# Interface automata

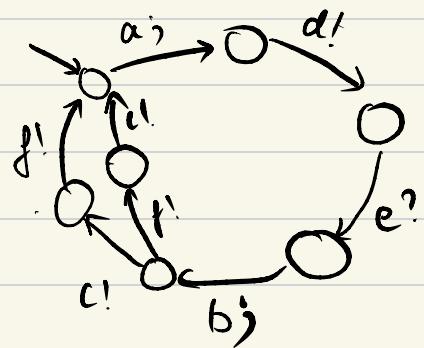
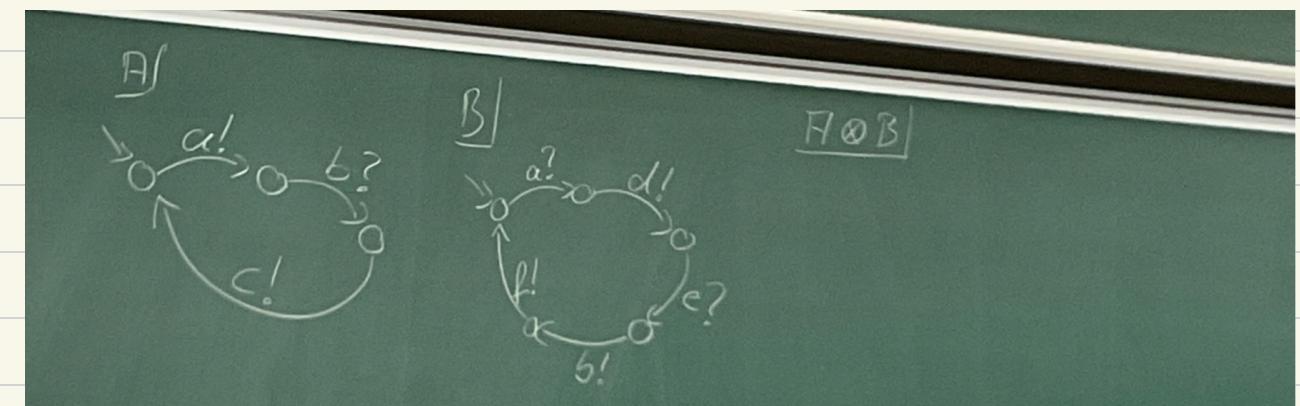
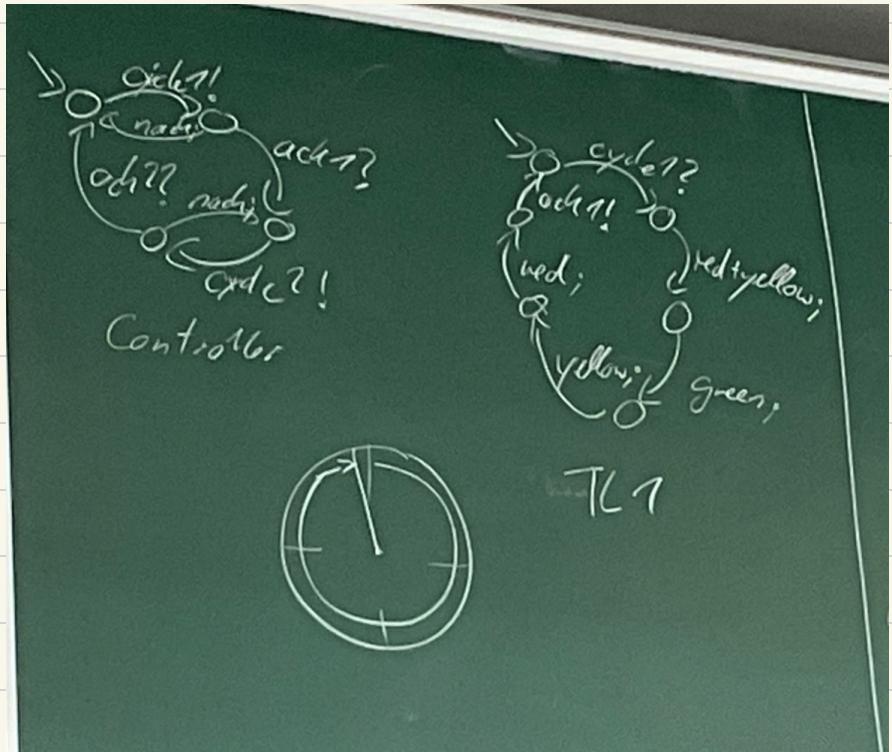
? input

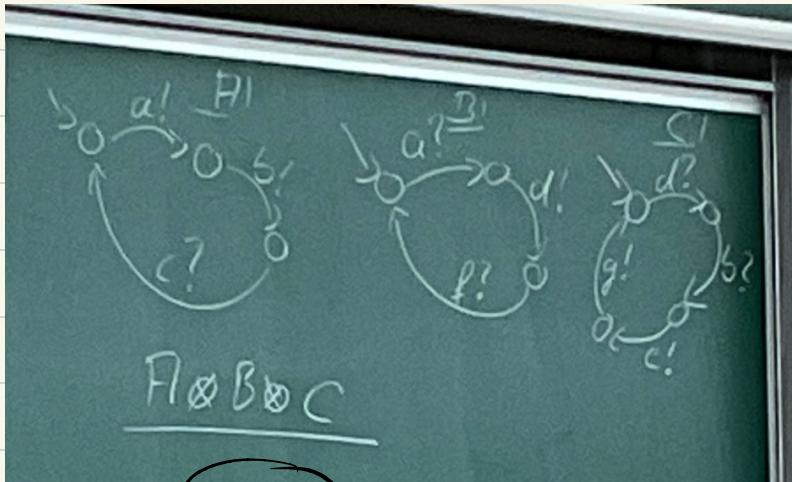
! output

Model a traffic light

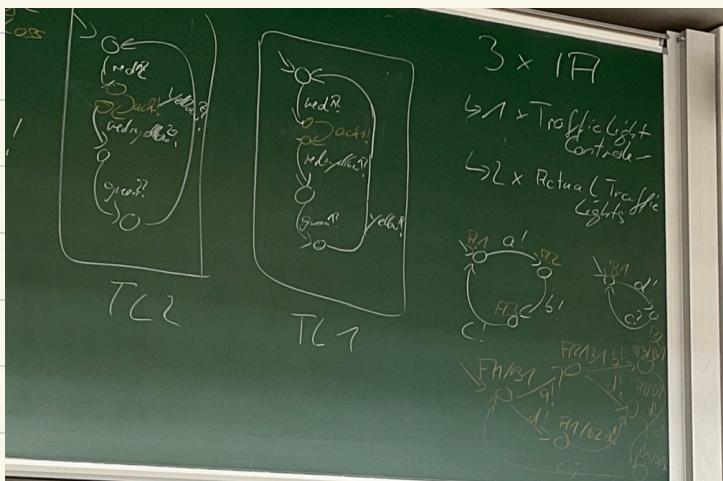
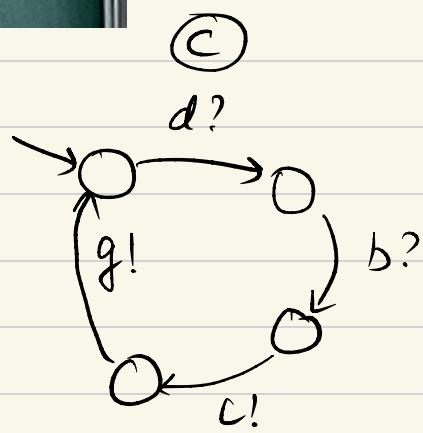
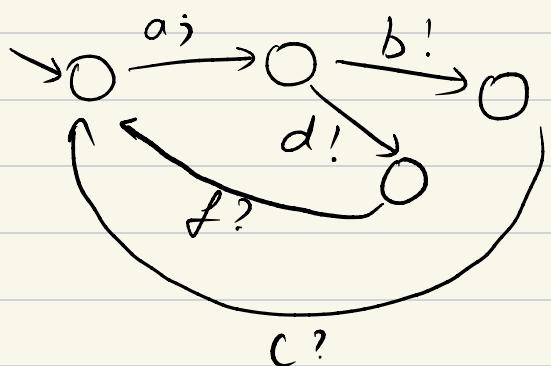
( 1 traffic control, 2 actual traffic lights)



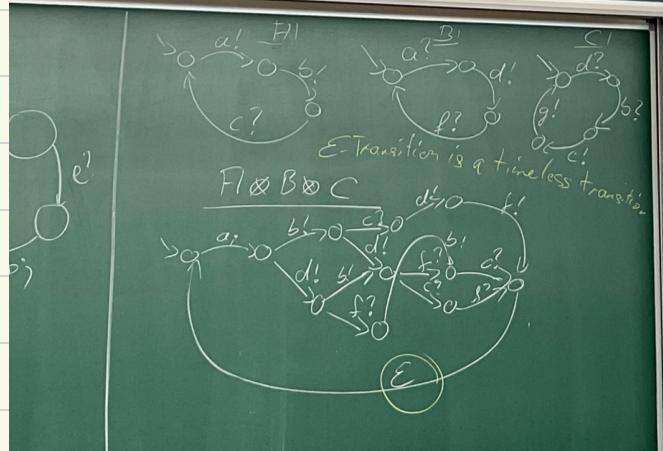


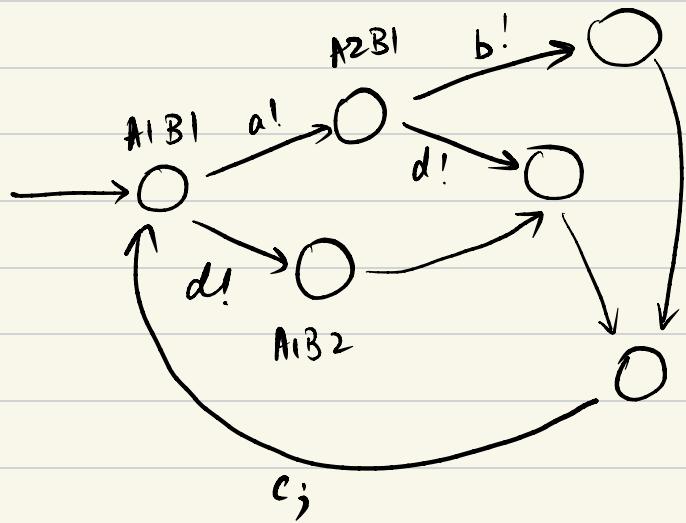
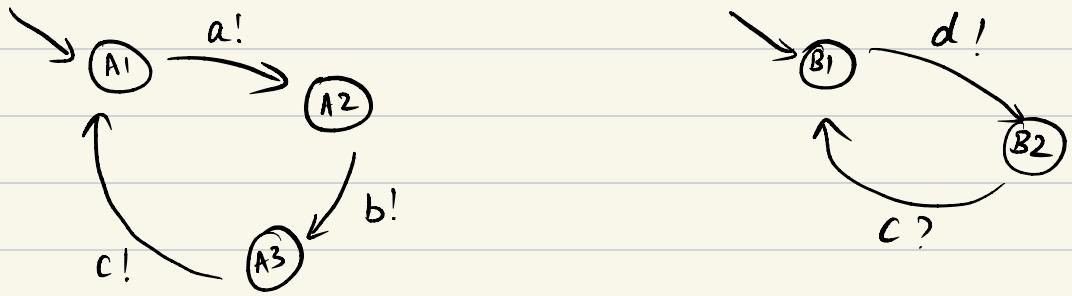


(A ⊗ B)



3 × 1F  
 ↳ 1 × Traffic Light Controller  
 ↳ 2 × Rectangular Traffic Lights





# Exercises Interface Automata

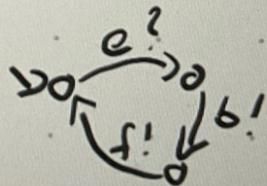
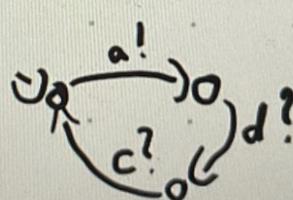
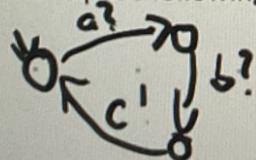
## Modelling

Access to a parking garage is controlled by gates. When a car approaches, the entrance gate checks with the garage management system whether a parking space is available. If this is the case, the gate opens and issues the parking ticket. Otherwise, the entrance gate will retry the check every 10 seconds.

Marian Daun  
When leaving the garage, the exit barrier reads the parking ticket and checks with the garage management system whether the parking ticket has been paid. If this is the case, the gate opens. If this is not the case, the barrier remains closed and the ticket is returned.

### Composition

Compose following IA



# Exercises Interface Automata

## Modelling

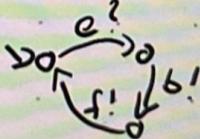
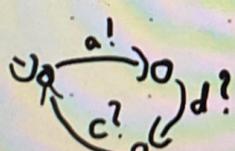
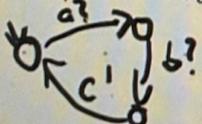
Access to a parking garage is controlled by gates. When a car approaches, the entrance gate checks with the garage management system whether a parking space is available. If this is the case, the gate opens and issues the parking ticket. Otherwise, the entrance gate will retry the check every 10 seconds. When leaving the garage, the exit barrier reads the parking ticket and checks with the garage management system whether the parking ticket has been paid. If this is the case, the gate opens. If this is not the case, the barrier remains closed and the ticket is returned.

Entrance Gate  
~~car at gate~~

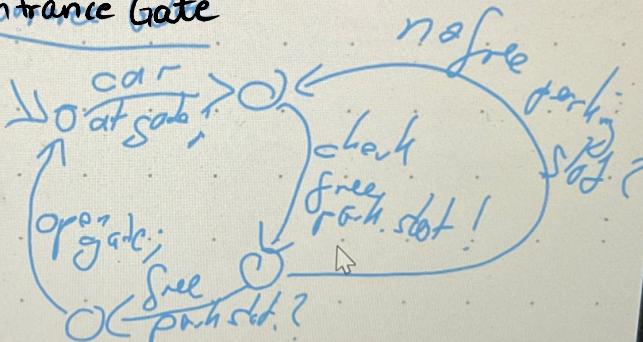
Marian Daun

### Composition

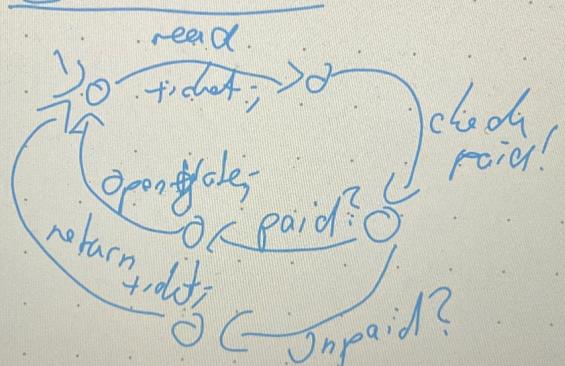
Compose following IA



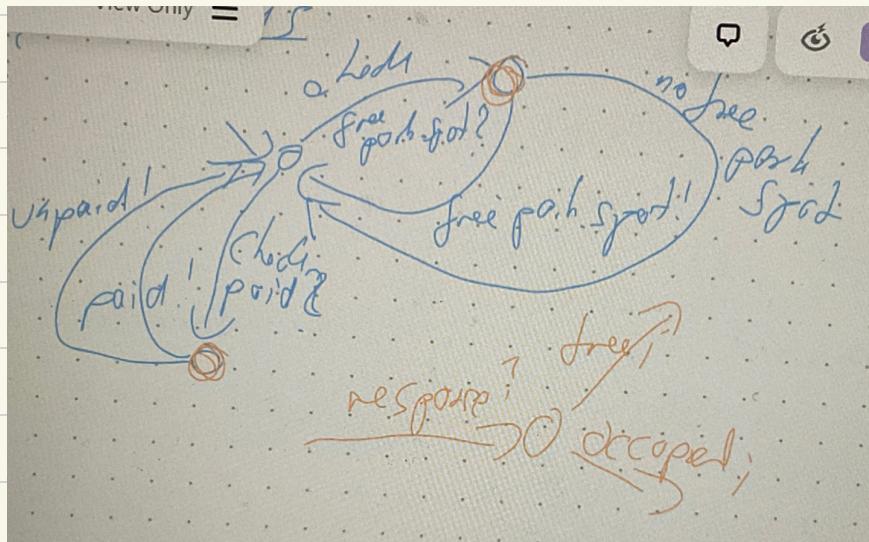
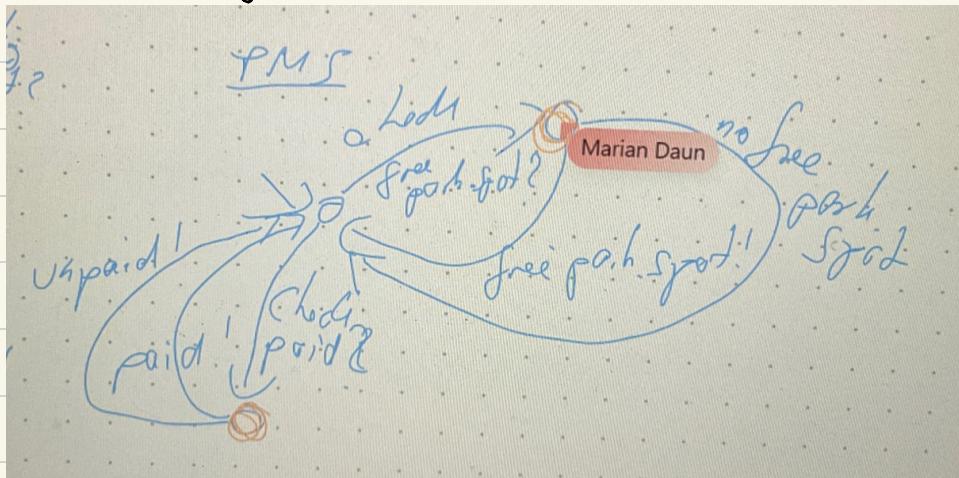
## Entrance Gate



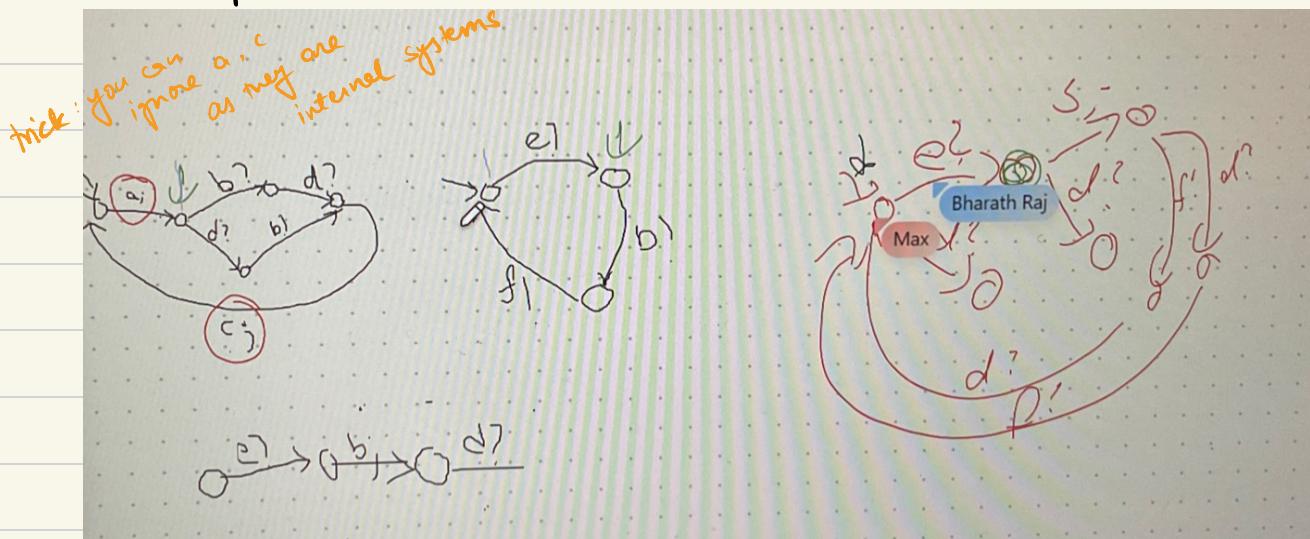
## Exit Barrier

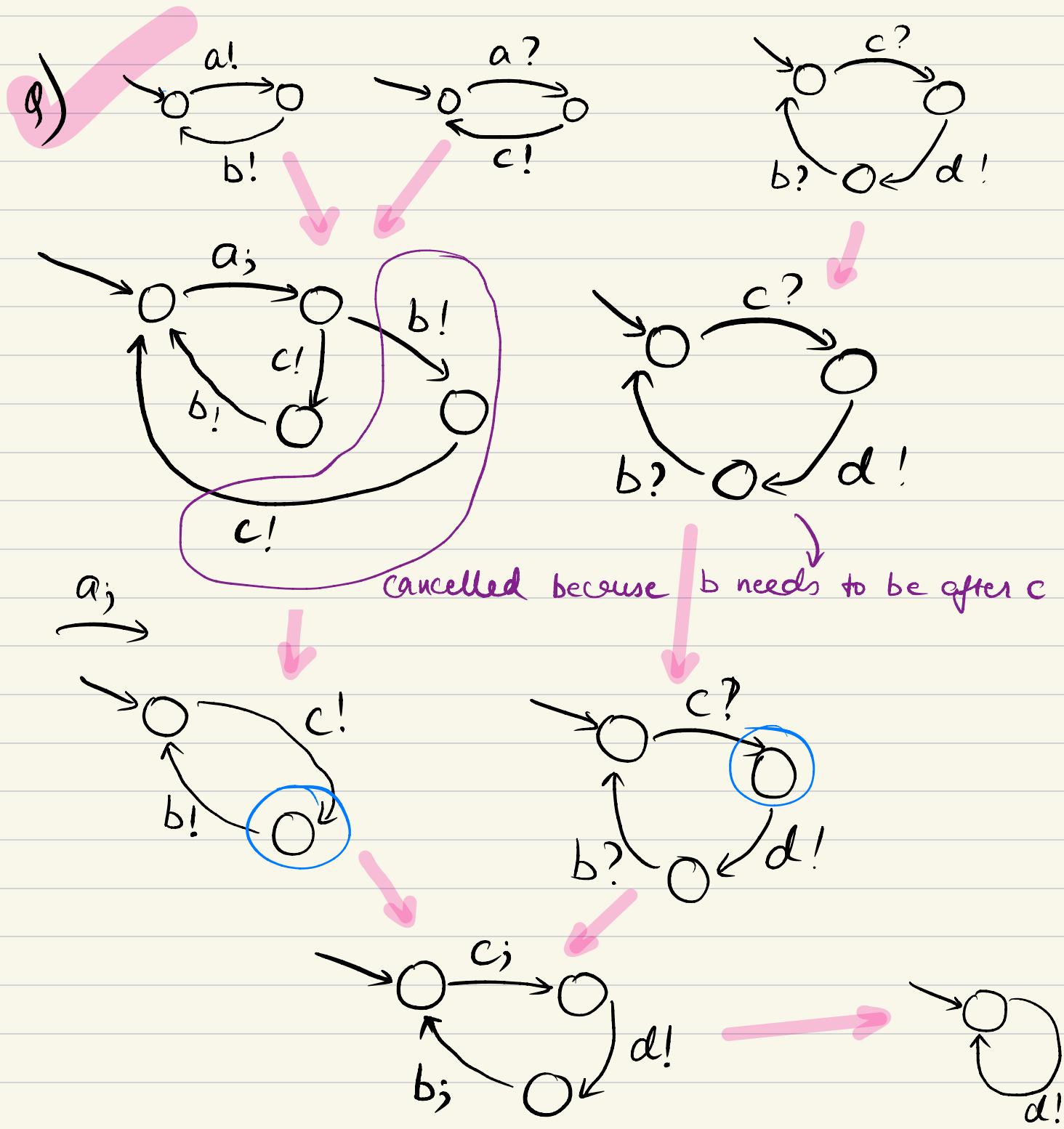
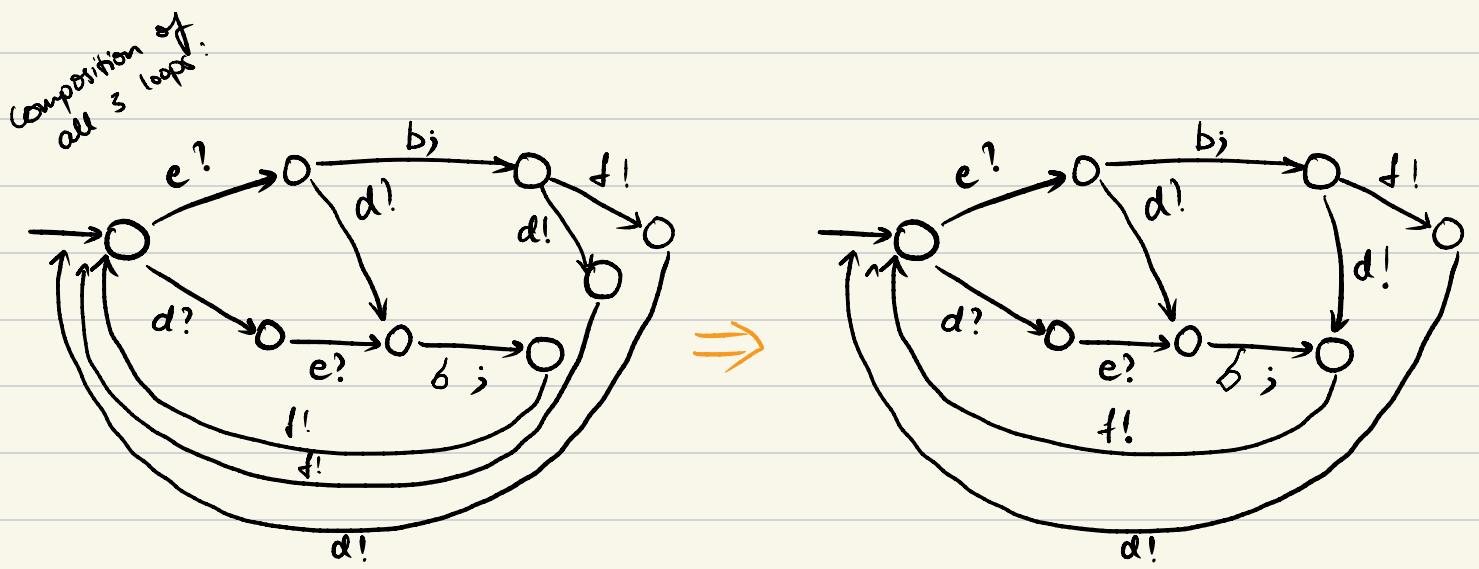


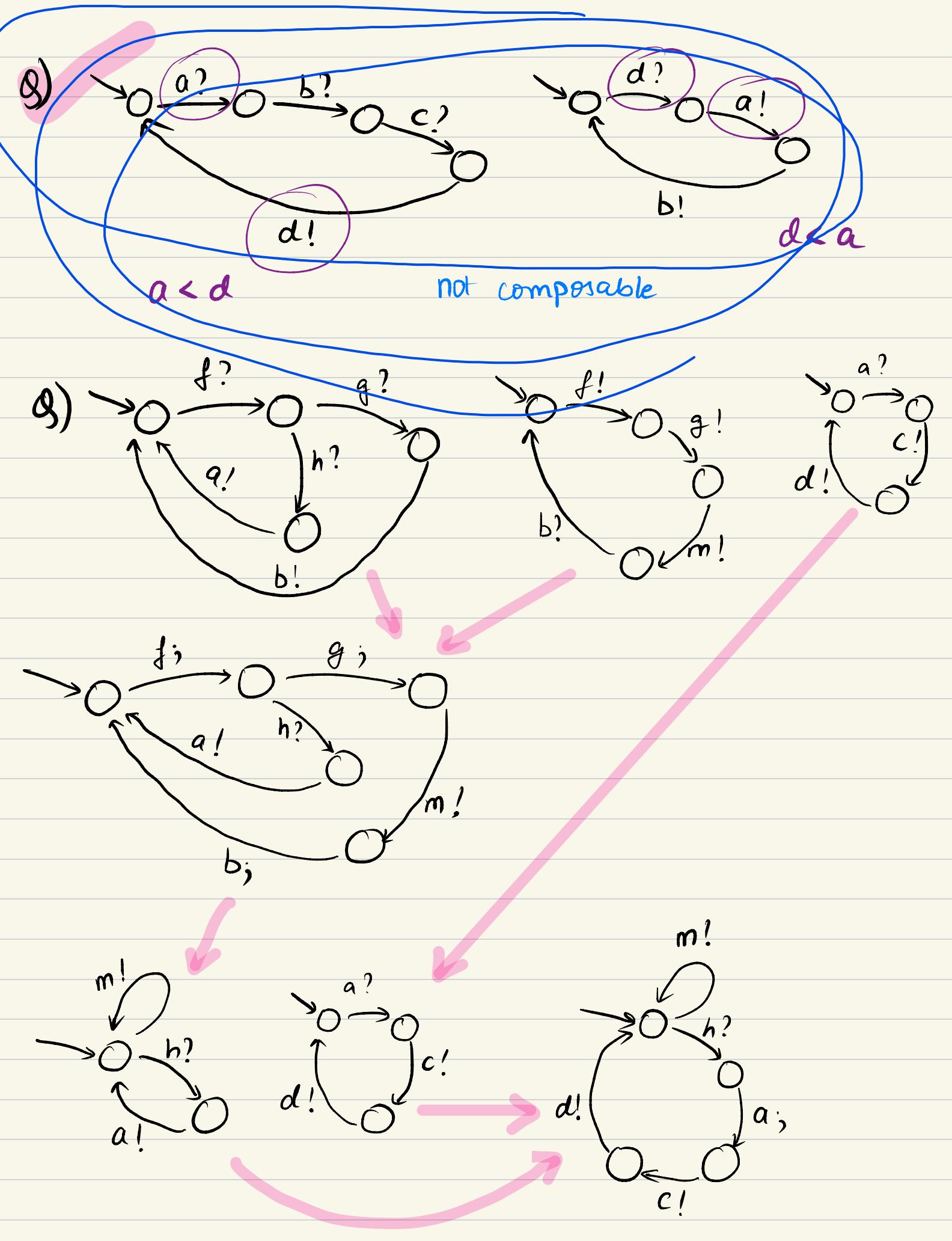
# parking management system



## Composition

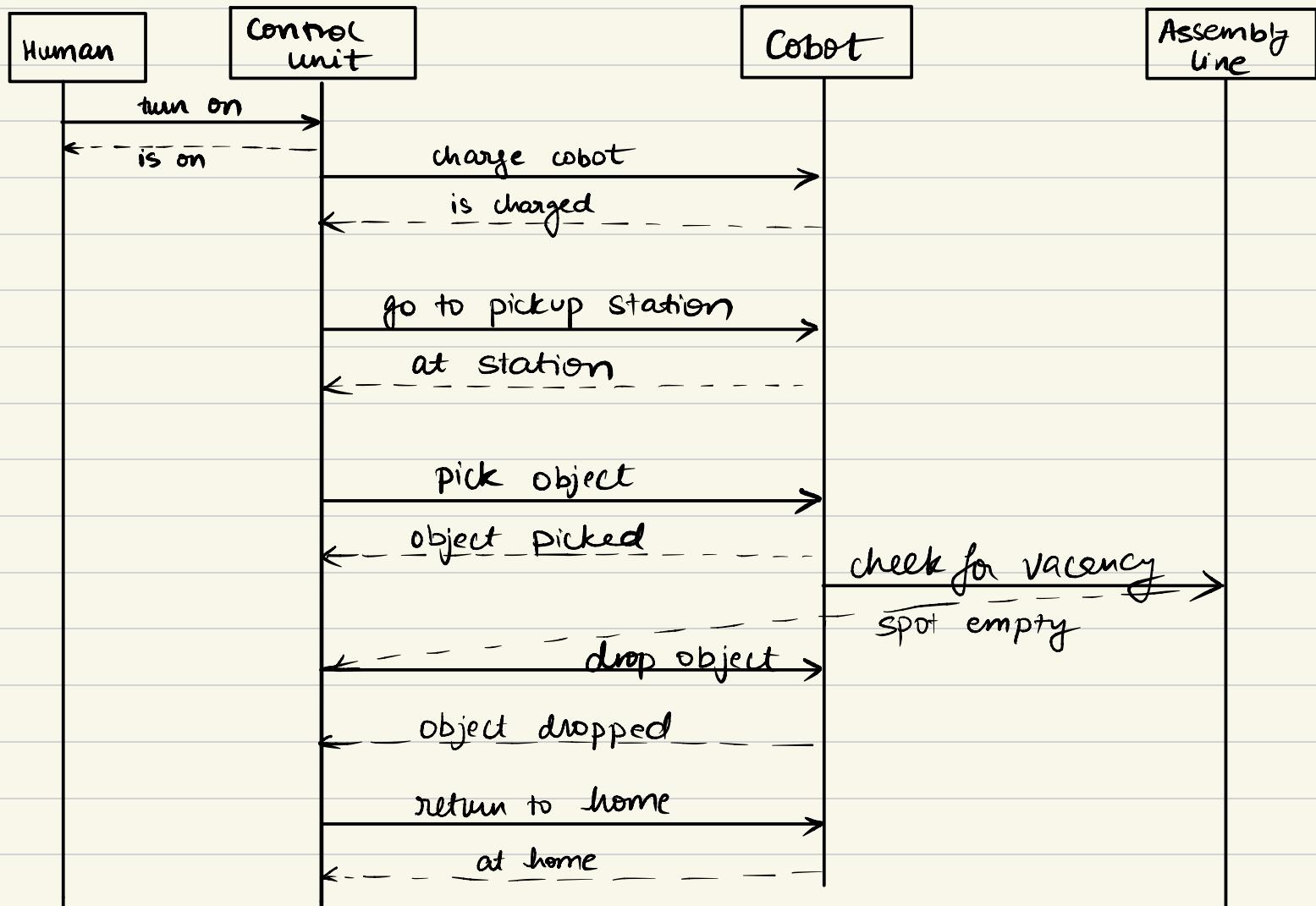








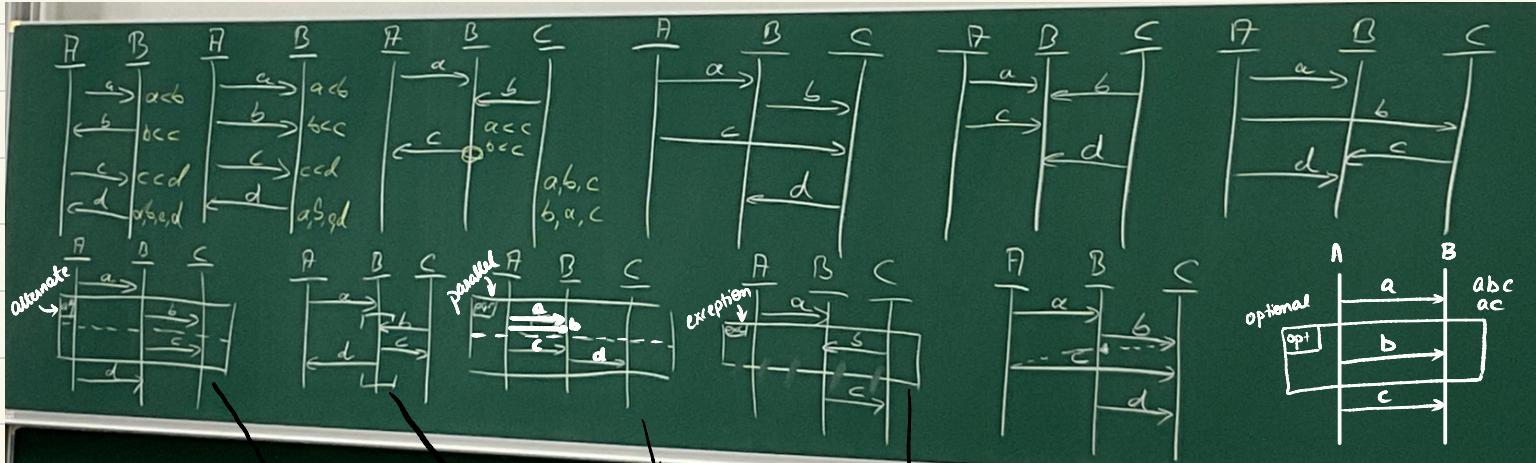
# Sequence Diagram of a Cobot



4)  $a < b$   
 $a < c$   
 $b < d$   
 $c < d$   
 $abcd$   
 $acbd$

5)  $a < c$   
 $b < d$   
 $acbd$   
 $abcd$   
 $abdc$   
 $bdac$

6)  $a < b$   
 $a < d$   
 $b < c$   
 $b < d$   
 $abcd$   
 $abdc$



7)  $a < b$   
 $a < d$   
 $a < c$

box has  
choice:  
above/  
dotted line

in this  
region  
order doesn't  
matter

both boxes  
can occur  
but you don't  
know which  
one

if b  
occurs, execution  
stops

abd  
acd  
adb  
adc

8)  $a < d$   
 $a < c$   
 $abed$     $bacd$   
 $abdc$     $badc$   
 $acbd$   
 $acdb$   
 $adbc$   
 $adcb$

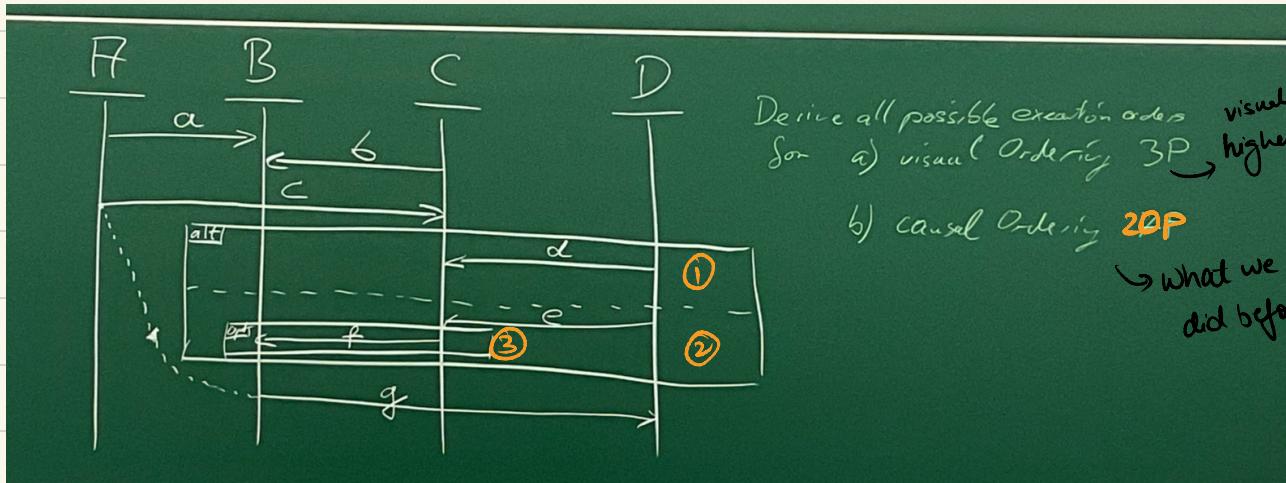
9)  $a < b$   
 $c < d$   
 $abcd$   
 $acbd$   
 $acdb$   
 $cdab$   
 $cadb$   
 $cabd$

10)  $a < c$   
 $ab$   
 $b$   
 $ac$

ii)  $a < d$   
 $a < c$   
 $b < d$   
 $b < c$

abcd  
abdc

Q)



Derive all possible execution orders visually  
from a) visual Ordering 3P → higher = corner first

b) causal Ordering 2DP

→ what we  
did before

a) abc dg  
abcef g  
abceg

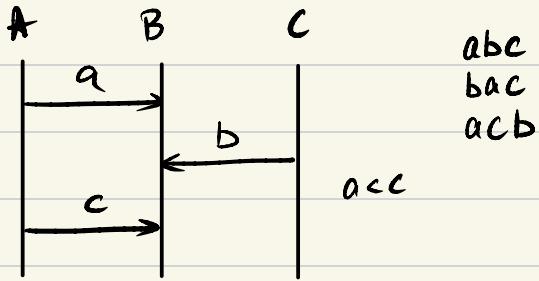
b)  $a < c$   
 $a < g$   
 $b < g$   
 $c < g$   
 $e < f$   
 $c < f$

① a, b, c, d, g  
a, c, b, d, g  
a, d, b, c, g  
a, d, c, b, g  
a, b, c, g, d  
a, c, b, f, d  
a, c, g, b, d

a, c, g, d, b  
a, c, d, g, b  
a, c, d, b, g  
b, a, c, d, g  
b, a, d, c, g  
b, a, c, g, d

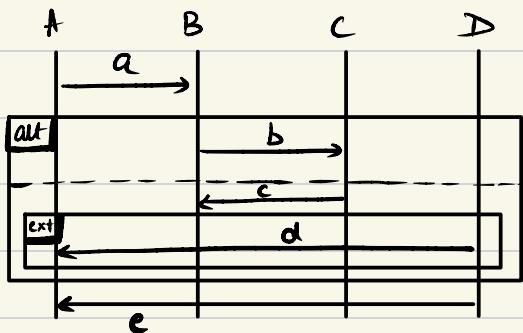
b, d, a, c, g  
d, a, b, c, g  
d, b, a, c, g  
d, a, c, b, g  
d, a, c, g, b

Q)



abc  
bac  
acb

Q)



a < b      abe      acb

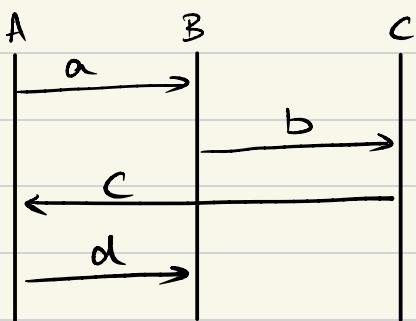
b < c      acd      d

d < e      ace      cd

cad      ad      cae

eac      cea      eca

Q)



Translate into IA

a < b      abcd

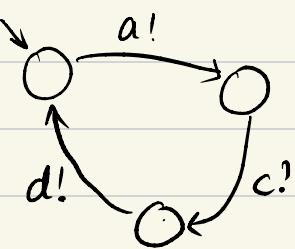
b < c

a < d

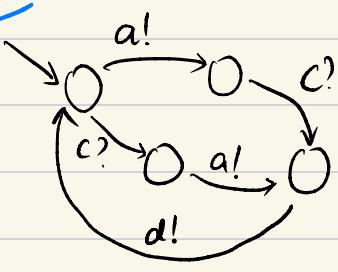
c < d

3 lifelines  $\Rightarrow$  3 IA

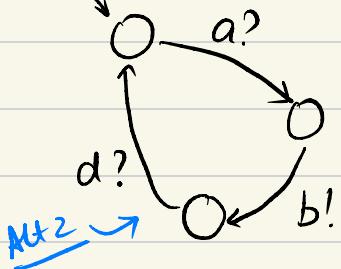
A]



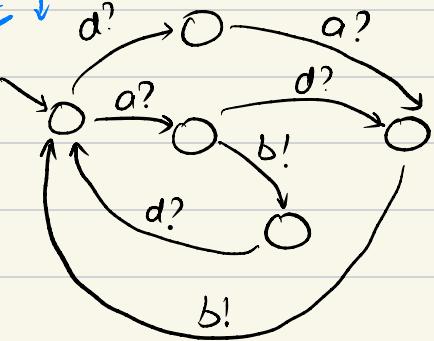
Alt 2  
Alt 1



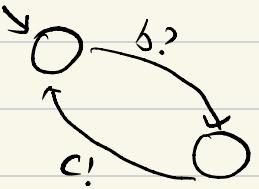
B]



Alt 2  
Alt 1

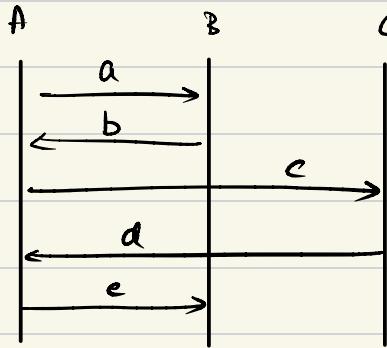


C]



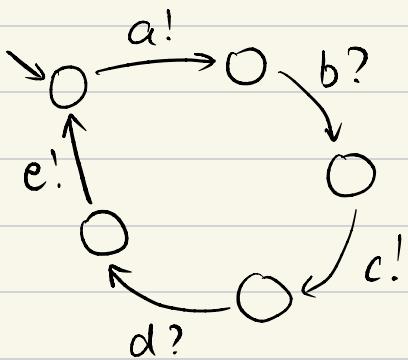
Alt 2  
Alt 1 is  
enough for the  
exam.

Q)

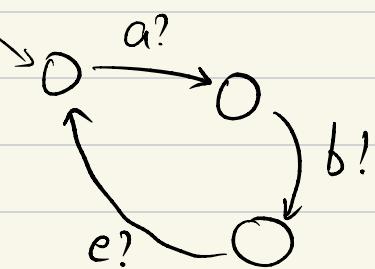


Translate into IA

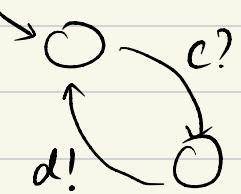
A]



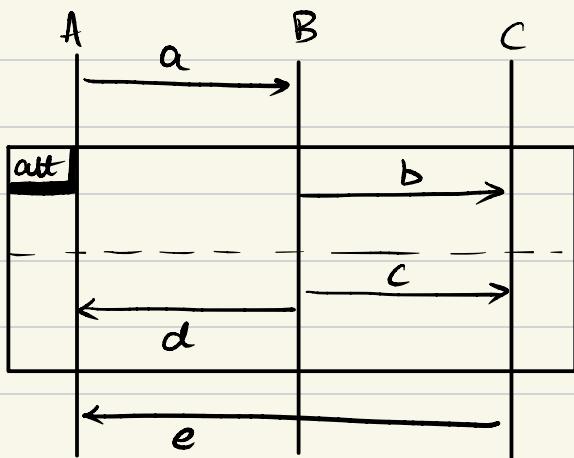
B]



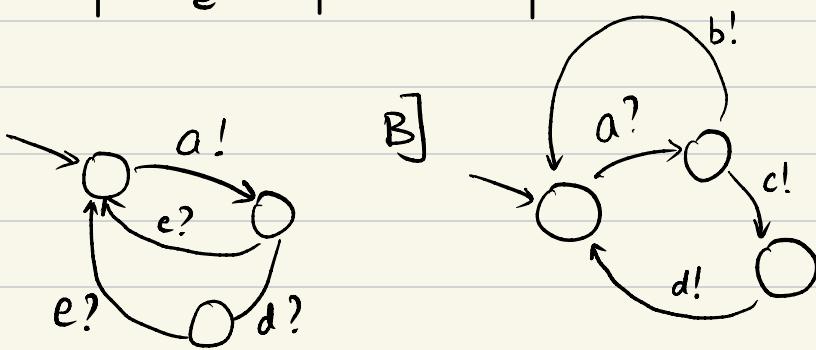
C]



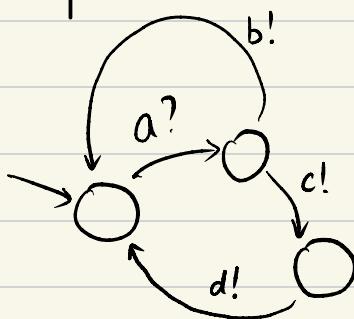
Q)



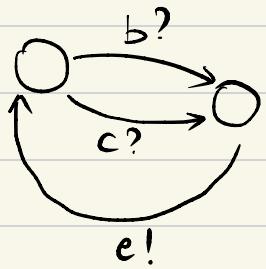
A]

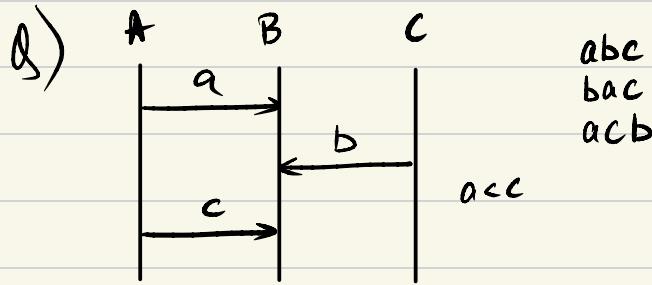


B]



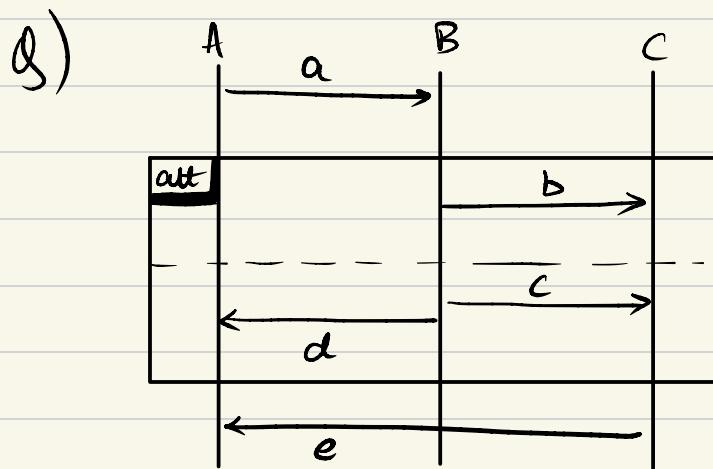
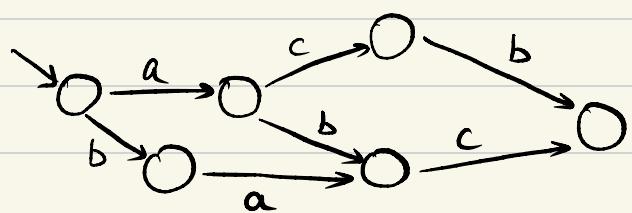
C]





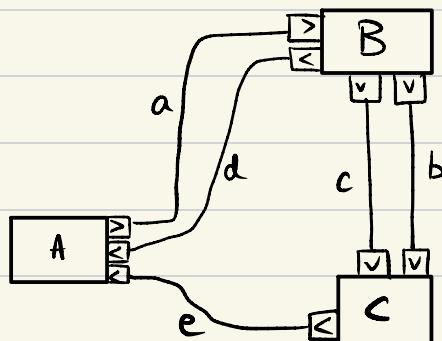
*finite state machine*

Transform into FSM



*internal block diagram*

Transform into IBD



## EMBEDDED SYSTEM

15 P System layers MAPE loop

15P {  
BDD  
IBD  
Content

$\frac{81}{90}$  for 1.0

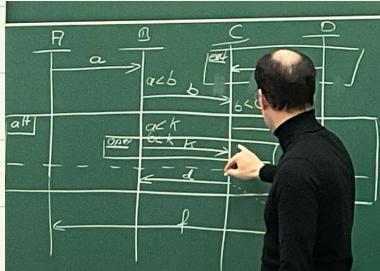
15 Petrinets Model | Analyse

15 IA Model | Compose | transform to SD

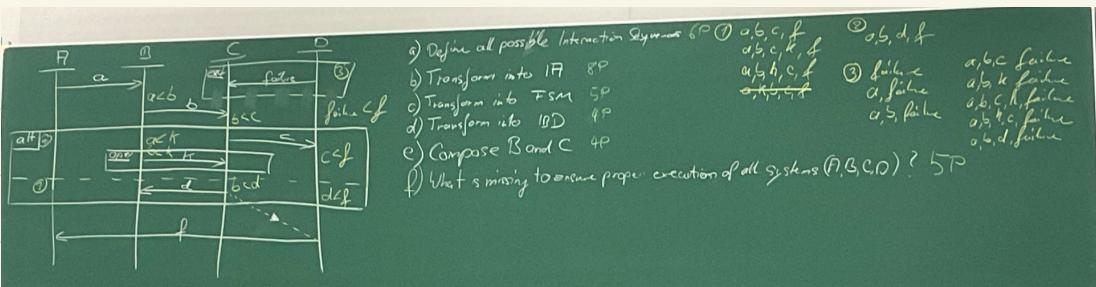
15 Sequence diagrams Model | Analysis | transform to IA, FSM, IBD

15 field buses

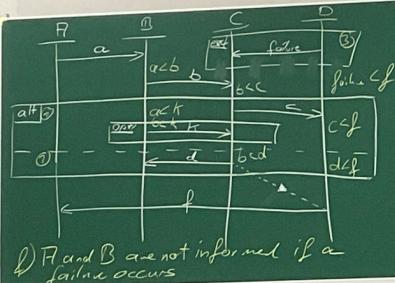
3 big modelling questions maybe + conversions



- a) Define all possible Interaction Diagrams 6P  
 b) Transform into IF 8P  
 c) Transform into FSM 5P  
 d) Transform into BD 4P  
 e) Compose B and C 4P  
 f) What is missing to ensure proper execution of all systems (A,B,C,D)? 5P

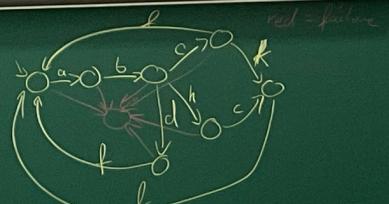
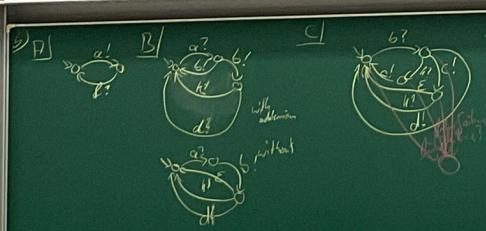
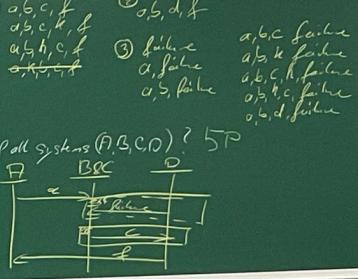


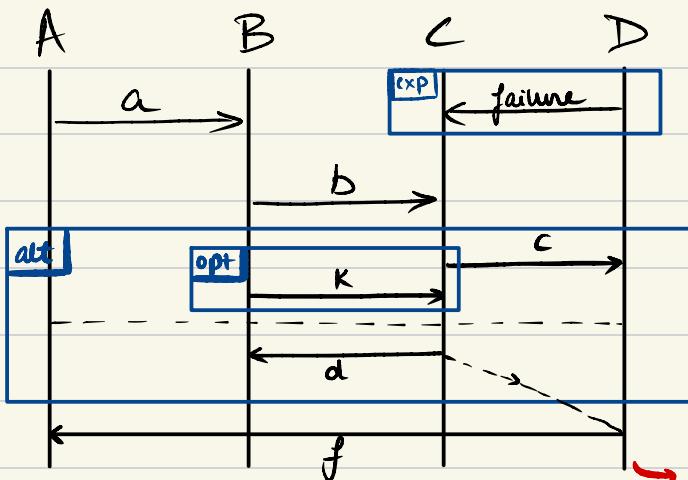
- a) Define all possible Interaction Diagrams 6P ① a,b,c,f  
 ② a,b,d,f  
 a,b,c,failure  
 a,b,k,failure  
 a,failure  
 a,b,c,h,failure  
 a,b,g,c,failure  
 a,c,d,failure  
 b,failure  
 a,b,k,failure  
 a,b,c,h,failure  
 a,b,g,c,failure  
 a,c,d,failure
- b) Transform into IF 8P ③ failure\_B  
 ④ failure\_C  
 ⑤ failure\_D  
 ⑥ failure\_A
- c) Transform into FSM 5P ⑦ a,b,c,f  
 ⑧ a,b,d,f  
 ⑨ a,b,c,failure  
 ⑩ a,b,k,failure  
 ⑪ a,failure  
 ⑫ a,b,c,h,failure  
 ⑬ a,b,g,c,failure  
 ⑭ a,c,d,failure
- d) Transform into BD 4P ⑮ a,b,c,f  
 ⑯ a,b,d,f  
 ⑰ a,b,c,failure  
 ⑱ a,b,k,failure  
 ⑲ a,failure  
 ⑳ a,b,c,h,failure  
 ㉑ a,b,g,c,failure  
 ㉒ a,c,d,failure
- e) Compose B and C 4P ㉓ a,b,c,f  
 ㉔ a,b,d,f  
 ㉕ a,b,c,failure  
 ㉖ a,b,k,failure  
 ㉗ a,failure  
 ㉘ a,b,c,h,failure  
 ㉙ a,b,g,c,failure  
 ㉚ a,c,d,failure
- f) What is missing to ensure proper execution of all systems (A,B,C,D)? 5P



f) A and B are not informed if a failure occurs

- g) Define all possible Interaction Diagrams 6P ① a,b,c,f  
 ② a,b,d,f  
 a,b,c,f,failure  
 a,b,k,f,failure  
 a,f,failure  
 a,b,c,h,f,failure  
 a,b,g,c,f,failure  
 a,c,d,f,failure
- h) Transform into IF 8P ③ failure\_B  
 ④ failure\_C  
 ⑤ failure\_D  
 ⑥ failure\_A
- i) Transform into FSM 5P ⑦ a,b,c,f  
 ⑧ a,b,d,f  
 ⑨ a,b,c,failure  
 ⑩ a,b,k,failure  
 ⑪ a,failure  
 ⑫ a,b,c,h,failure  
 ⑬ a,b,g,c,failure  
 ⑭ a,c,d,failure
- j) Transform into BD 4P ⑮ a,b,c,f  
 ⑯ a,b,d,f  
 ⑰ a,b,c,failure  
 ⑱ a,b,k,failure  
 ⑲ a,failure  
 ⑳ a,b,c,h,failure  
 ㉑ a,b,g,c,failure  
 ㉒ a,c,d,failure
- k) Compose B and C 4P ㉓ a,b,c,f  
 ㉔ a,b,d,f  
 ㉕ a,b,c,failure  
 ㉖ a,b,k,failure  
 ㉗ a,failure  
 ㉘ a,b,c,h,failure  
 ㉙ a,b,g,c,failure  
 ㉚ a,c,d,failure
- l) What is missing to ensure proper execution of all systems (A,B,C,D)? 5P





$a < k$

$a < b$

$b < c$

$b < k$

$\text{failure} < f$

$d < f$

$b < d$

$c < f$

only look at D for failure order.  
failure occurs before f. BUT  
c and d don't have any  
order w.r.t failure

### 1) Define all possible Interaction system (6P)

abcf failure

abckf a failure

abkcfa failure

abc failure

abck failure

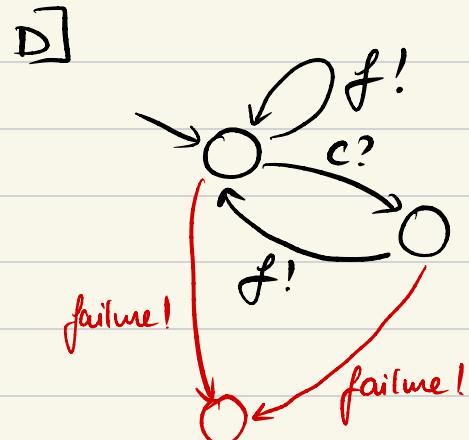
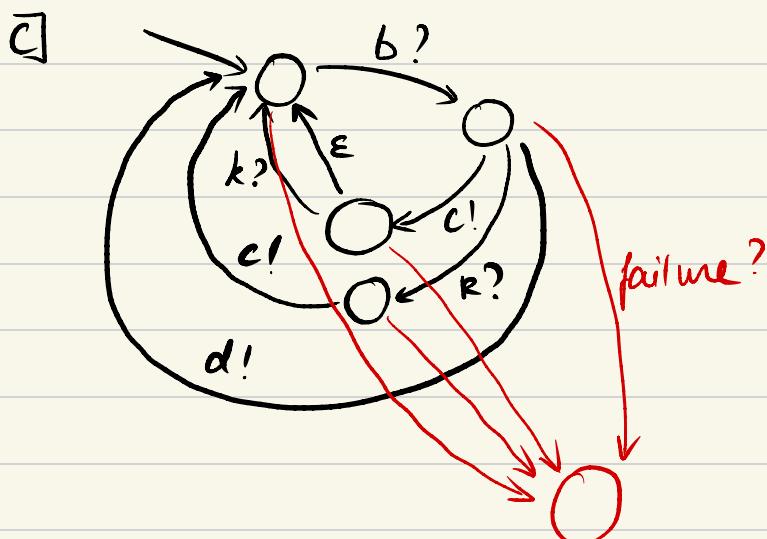
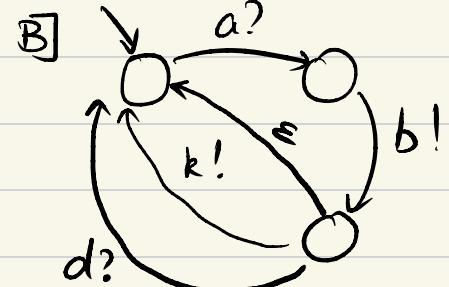
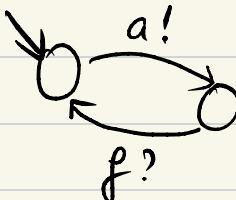
abkc failure

abk failure

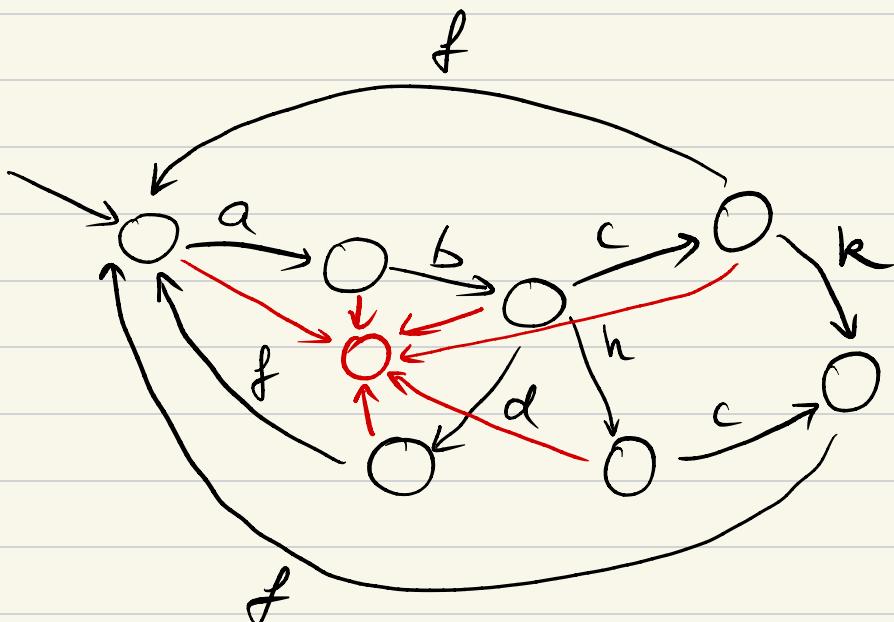
abdf

abd failure

### 2) Transform into IA

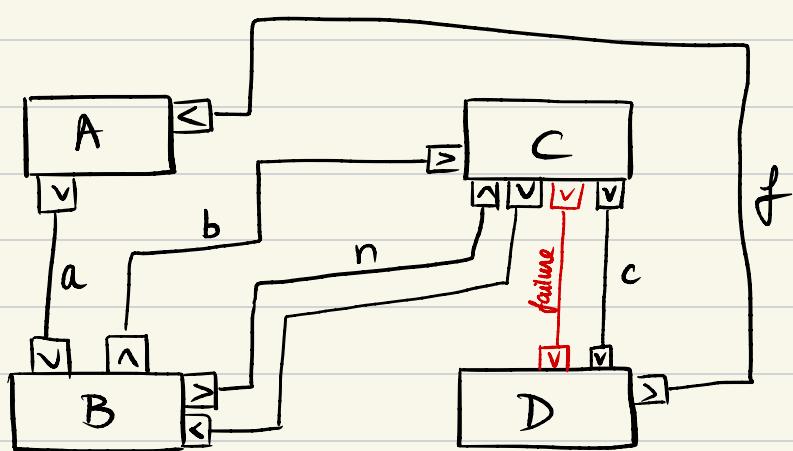


3)

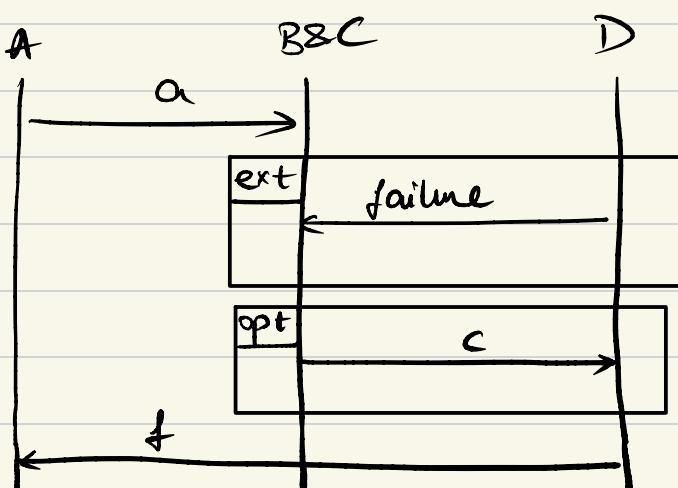


red = failure

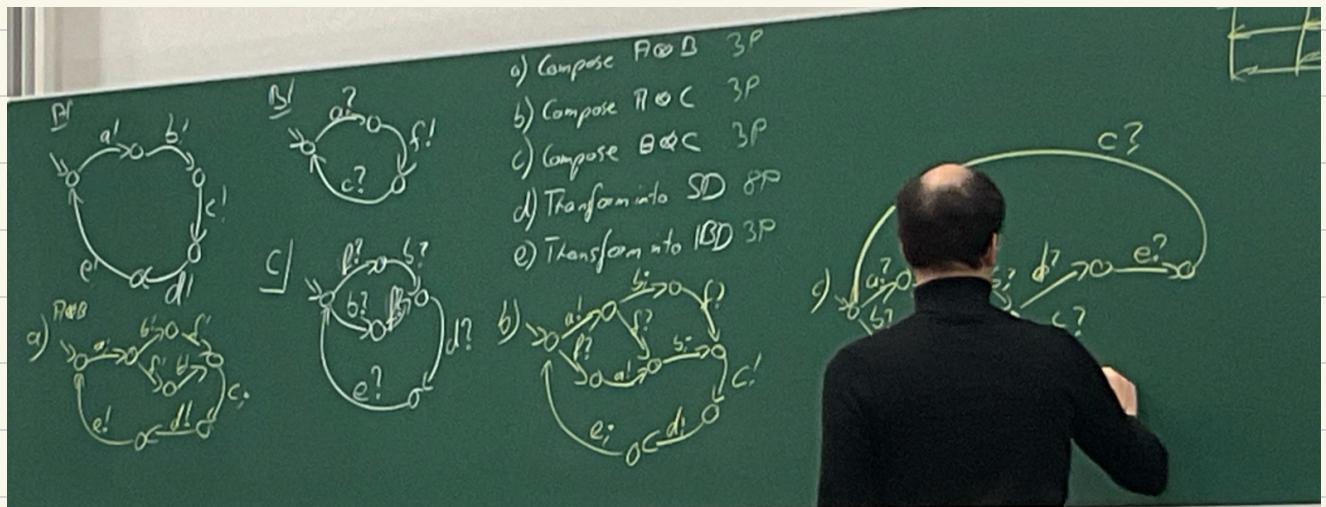
4)

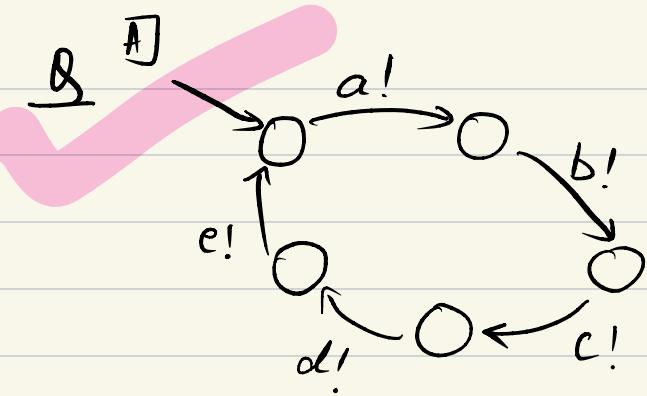


5)

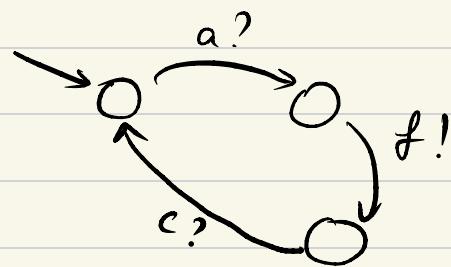


6) A and B are not informed if a failure occurs.

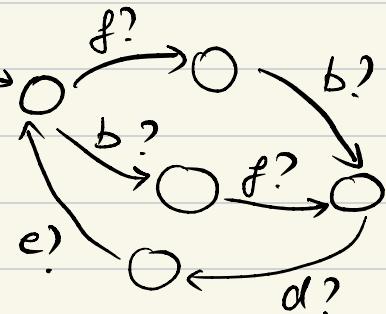




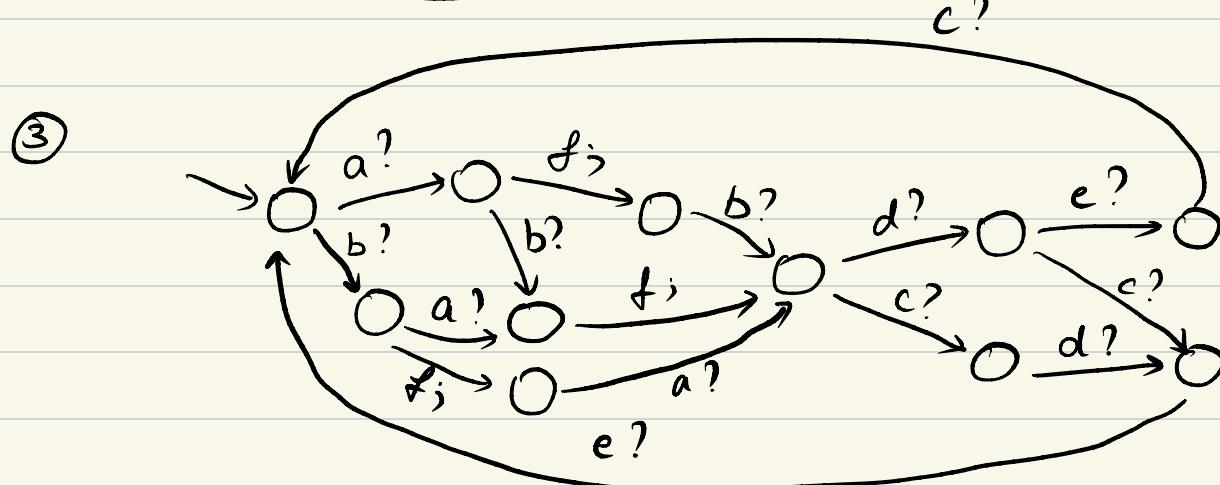
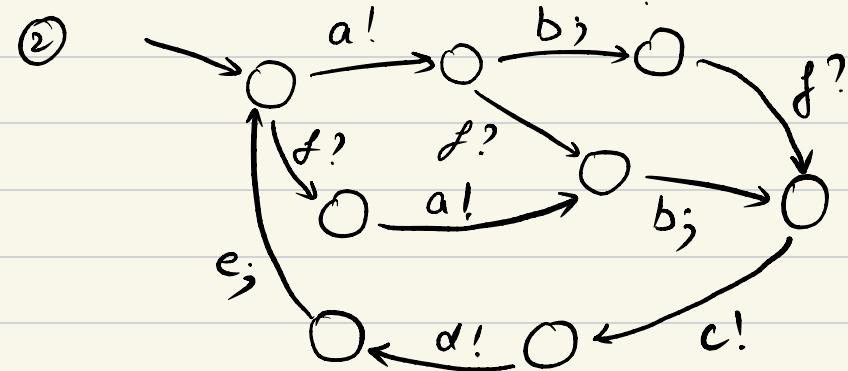
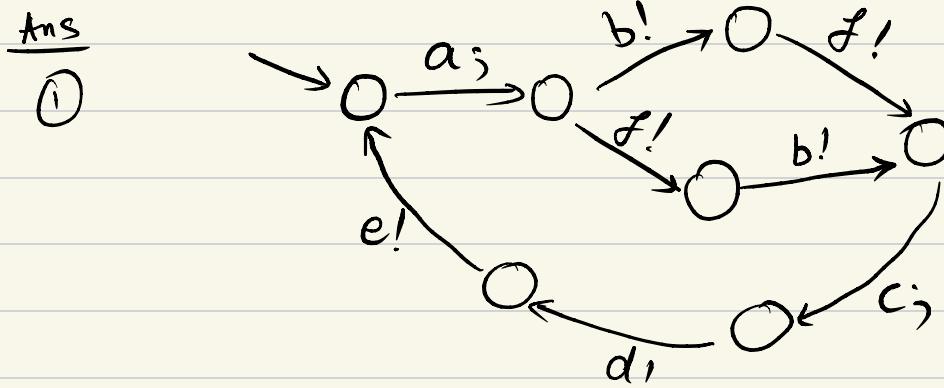
**B**

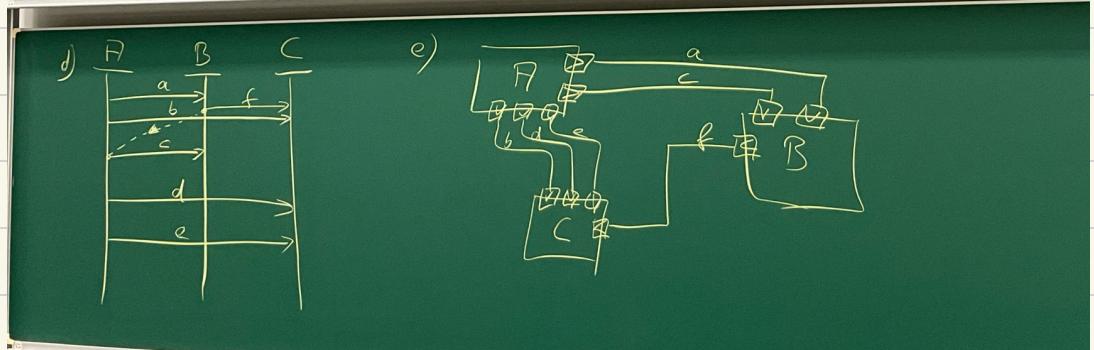
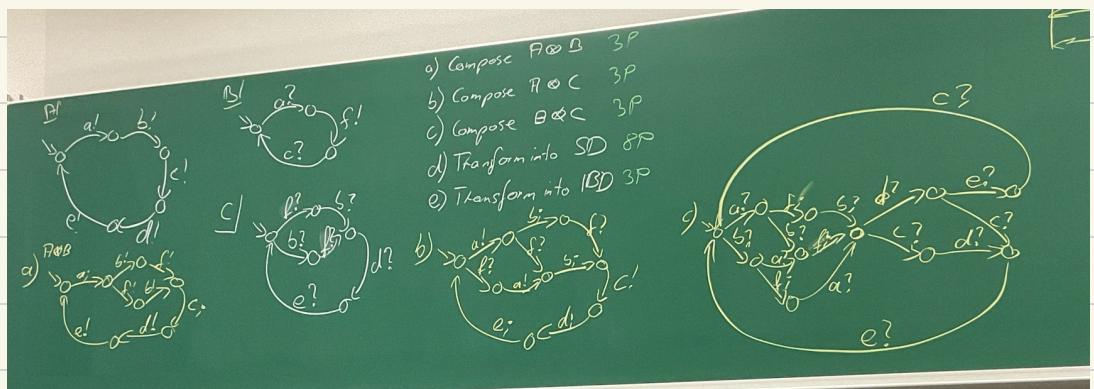


**C**

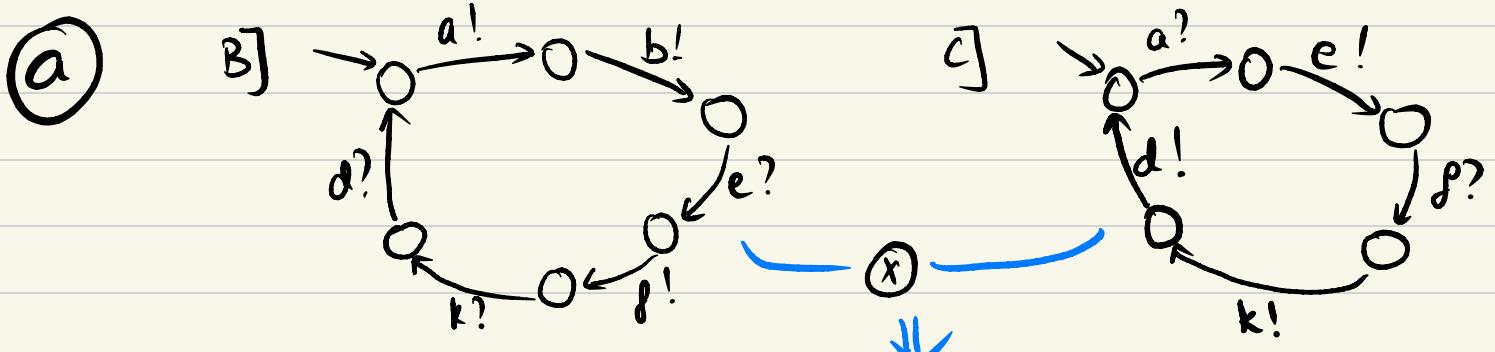
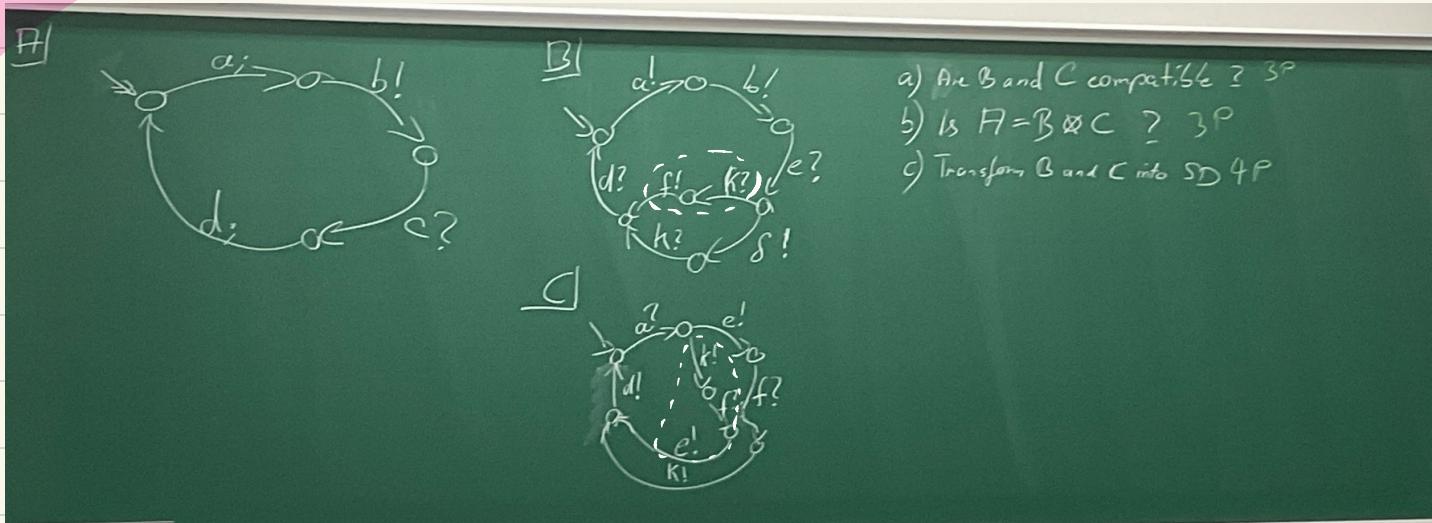


- ① Compose  $A \otimes B$
- ② Compose  $A \otimes C$
- ③ Compose  $B \otimes C$
- ④ Transform into SD
- ⑤ Transform into IBD



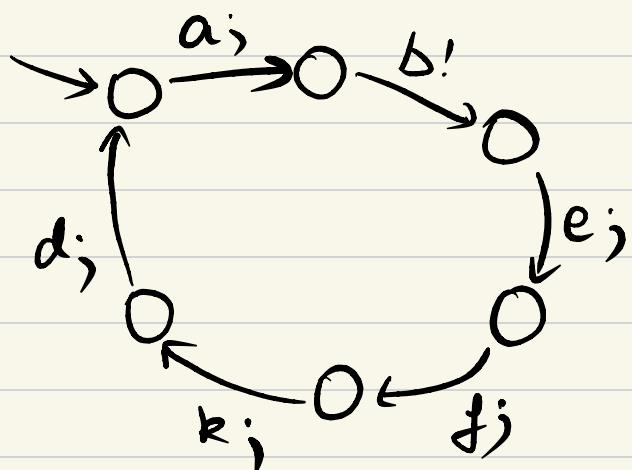


Q)

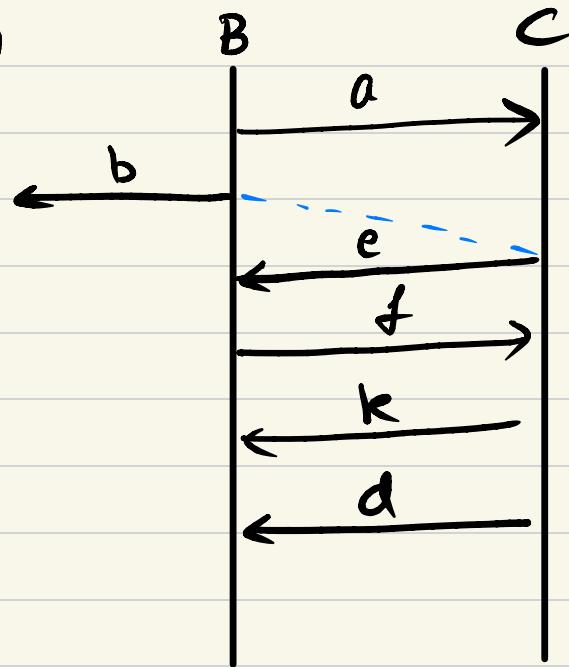


**B**  $\otimes$  **C**

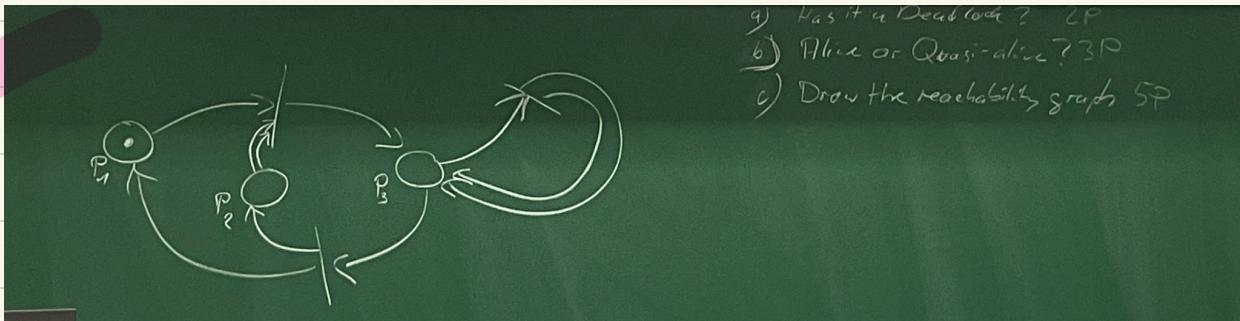
$\rightarrow \circlearrowleft b!$



**b)** No  $\neq B \otimes C$



Q)



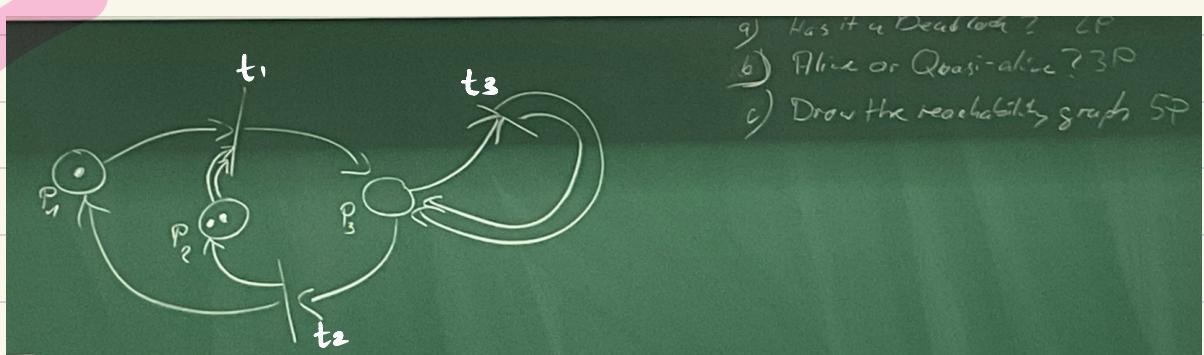
- g) Has it a Deadlock? 2P  
b) Alive or Quasi-alive? 3P  
c) Draw the reachability graph 5P

a) Fully dead - it has a deadlock

b) Dead -

c) 100

Q)



- g) Has it a Deadlock? 2P  
b) Alive or Quasi-alive? 3P  
c) Draw the reachability graph 5P

a) Yes, after  $t_1, t_2$

b) No, it will be dead

if there's even one possibility to reach a dead loop, its dead, not quasialive

c)  $(1, 2, 0)$  impossible due to infinity

$\downarrow$                                    $\uparrow$                                    $\uparrow$   
 $(0, 0, 1) \rightarrow (0, 0, 2) \rightarrow (0, 0, 3) \rightarrow (0, 0, 4) \rightarrow \dots$

$\downarrow$                                    $\downarrow$

$(1, 1, 0) \rightarrow (1, 1, 1) \rightarrow (1, 1, 2) \rightarrow (1, 1, 3) \rightarrow \dots$

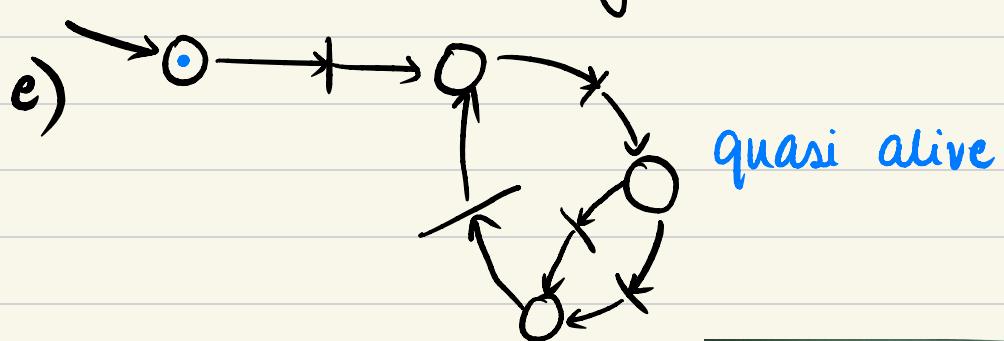
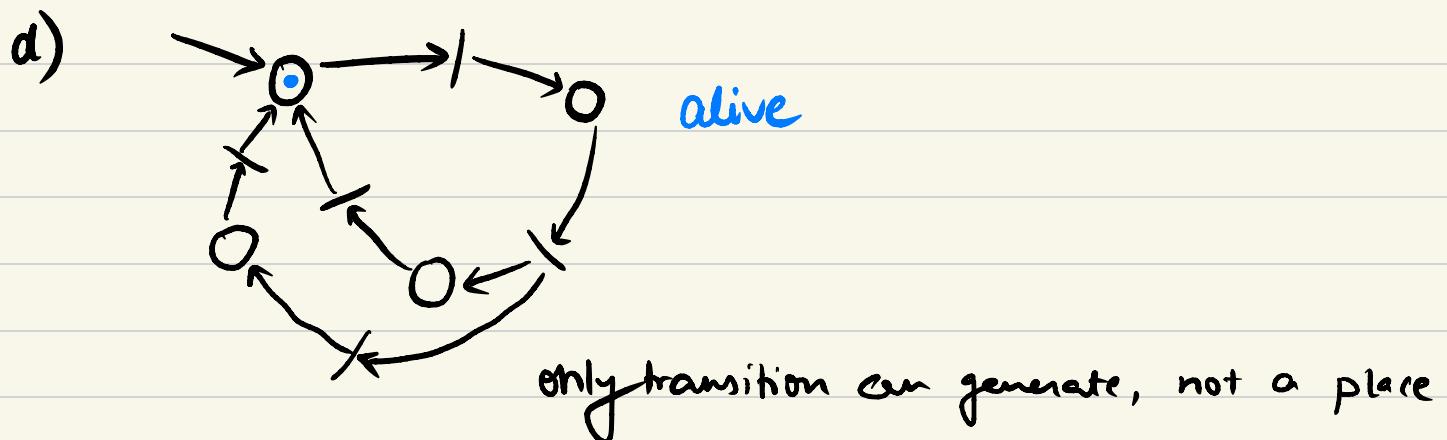
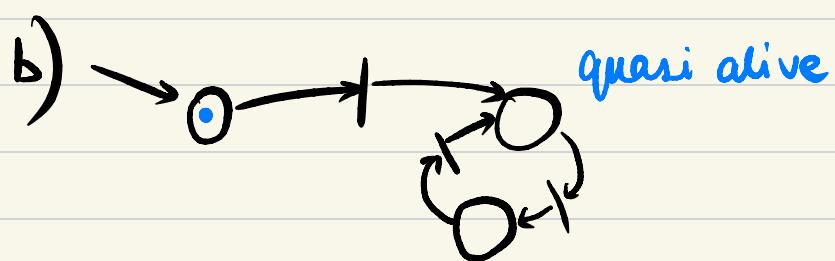
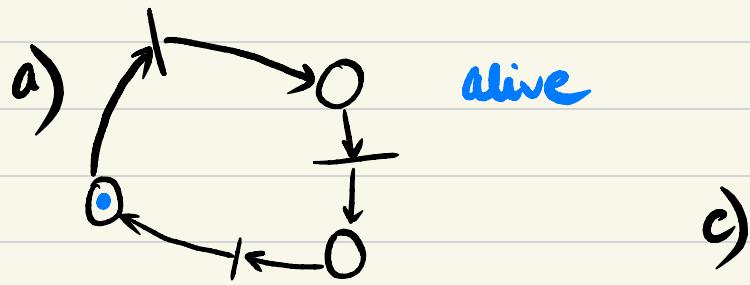
$\downarrow$                                    $\downarrow$                                    $\downarrow$                                    $\downarrow$

$(2, 2, 0) \dots \dots \dots \dots$

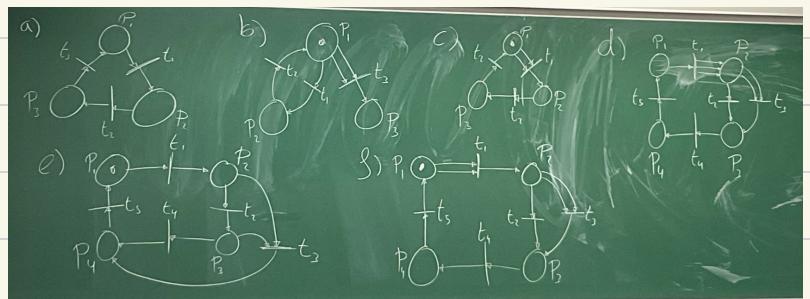
$\downarrow$                                    $\uparrow$

$(1, 0, 1) \rightarrow (1, 0, 2) \rightarrow (1, 0, 3) \rightarrow \dots$

Q) Develop a Petri Net with  
 $\frac{3}{4}$  places and  $\frac{3}{5}$  transitions that is  
 a) alive  
 b) quasi alive  
 c) neither



f)



## Exam Preparation

### Exam Tasks needed:

#### Petri Nets

- ↳ Model a Petri Net
- ↳ Check for Deadlocks
- ↳ Decide Liveness
- ↳ Derive Transition Graph

#### Interface Automata

- ↳ Model a set of IA
- ↳ Decide Compatibility
- ↳ Compose
- ↳ Transform to SD
- ↳ Transform to IBD

#### Field Buses

#### Sequence Diagrams

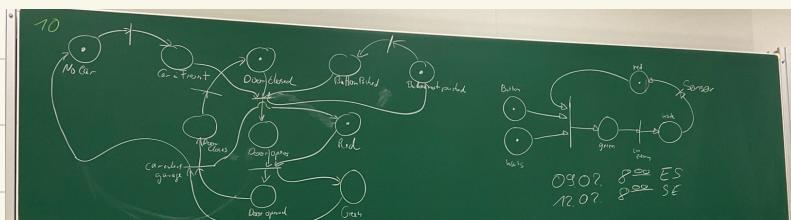
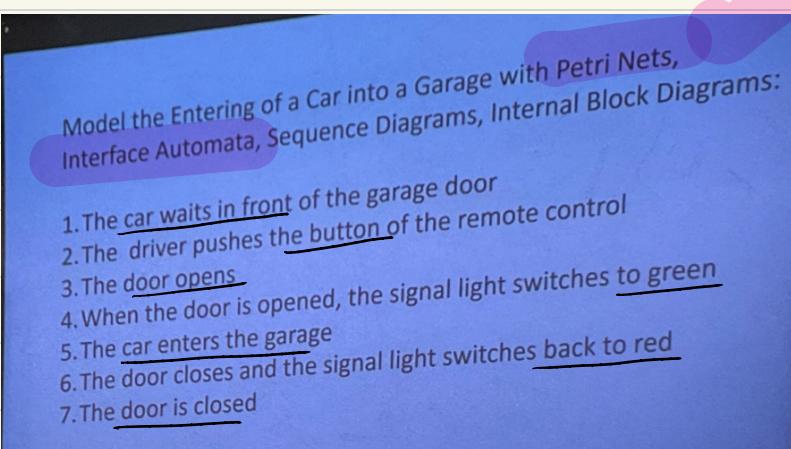
- ↳ Model a Sequence Diagram
- ↳ Decide Execution Orders
- ↳ Transform to IF
- ↳ Transform to FSM
- ↳ Transform to IBD
- ↳ Compose Components

#### Structure

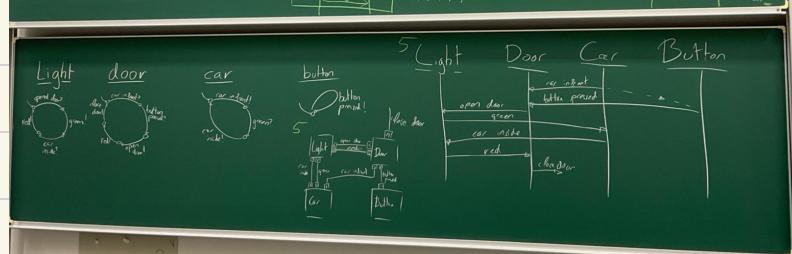
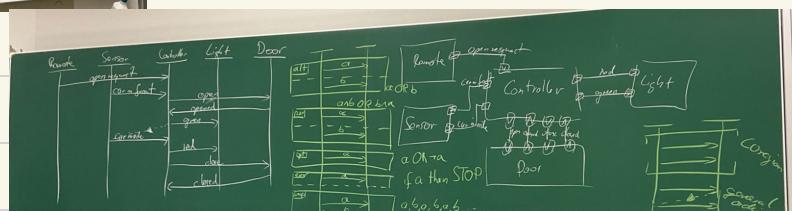
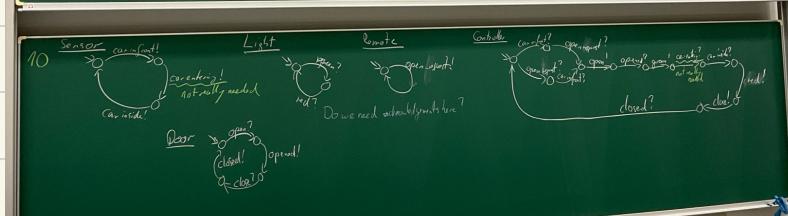
- ↳ Model BDD
- ↳ Model BD

#### Theory Questions

- ↳ MAPE
- ↳ Safety / Security
- ↳ ...



OB0? 8 ES  
OB1? 8 SE





## Exercise

We have a velocity sensor measuring the velocity of a wheel, calculating the resulting vehicle's speed, and transmitting the result via field bus.

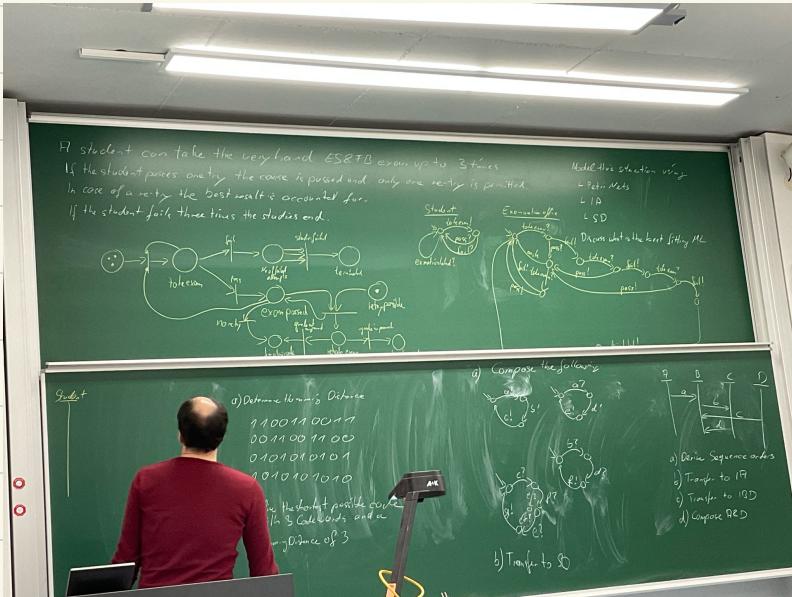
What do we need to consider when designing an ACC relying on the sensor?

- accuracy and precision of the sensor
- reliability
- low noise, low reflection
- good shielding
- good environment / weather conditions
- low cost
- sensor placement and response time / real time constraints
- cable arrangement
- mode of data transmission

A student can take the very hard ES8FB exam up to 3 times  
 If the student passes one try, the course is passed and only one retry is permitted.  
 In case of a rety, the best result is accounted for.  
 If the student fails three times the studies end.

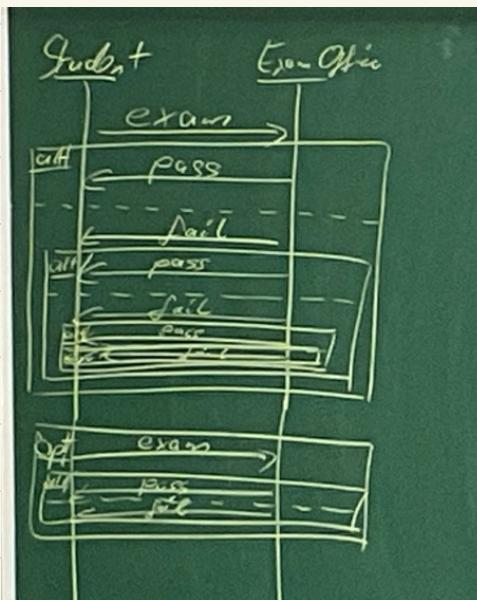
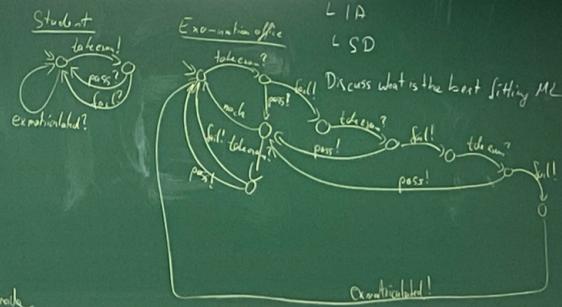
Model this situation using  
 L Petri Nets  
 L IA  
 L SD

Discuss what is the best fitting ML

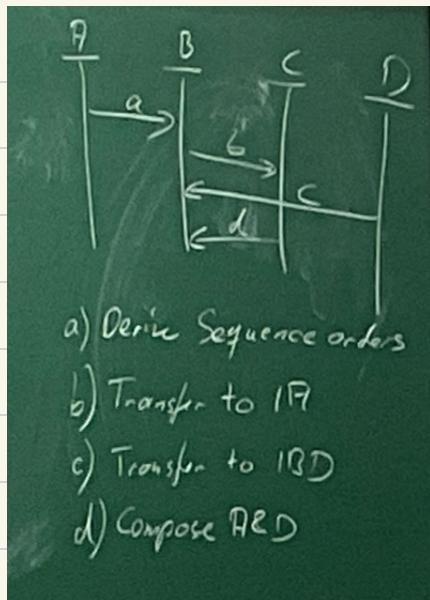


A student can take the very hard ES8FB exam up to 3 times  
 If the student passes one try, the course is passed and only one rety is permitted.  
 In case of a rety, the best result is accounted for.  
 If the student fails three times the studies end.

Model this situation using  
 L Petri Nets  
 L IA  
 L SD

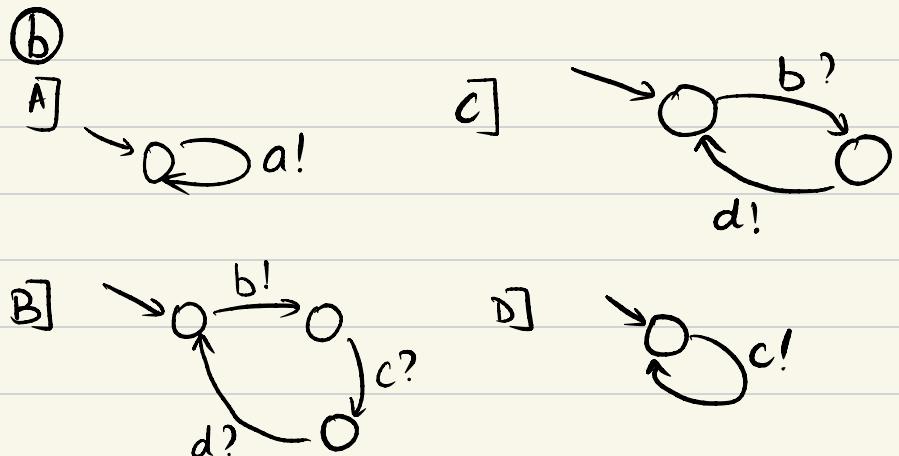


8

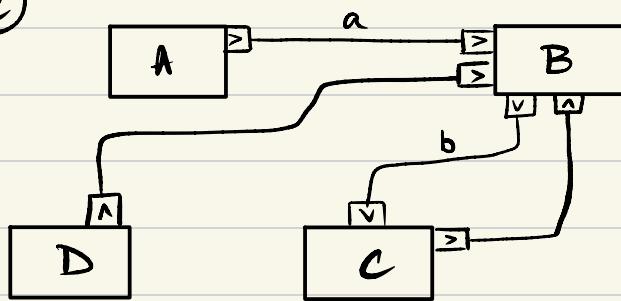


$a < b$       @ abcd  
 $b < c$       acbd  
 $c < d$       cabd

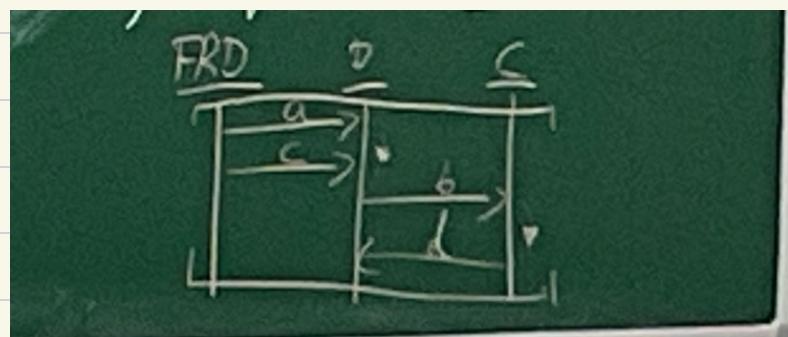
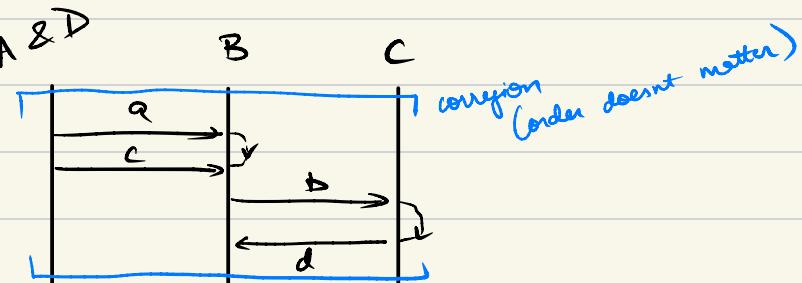
④

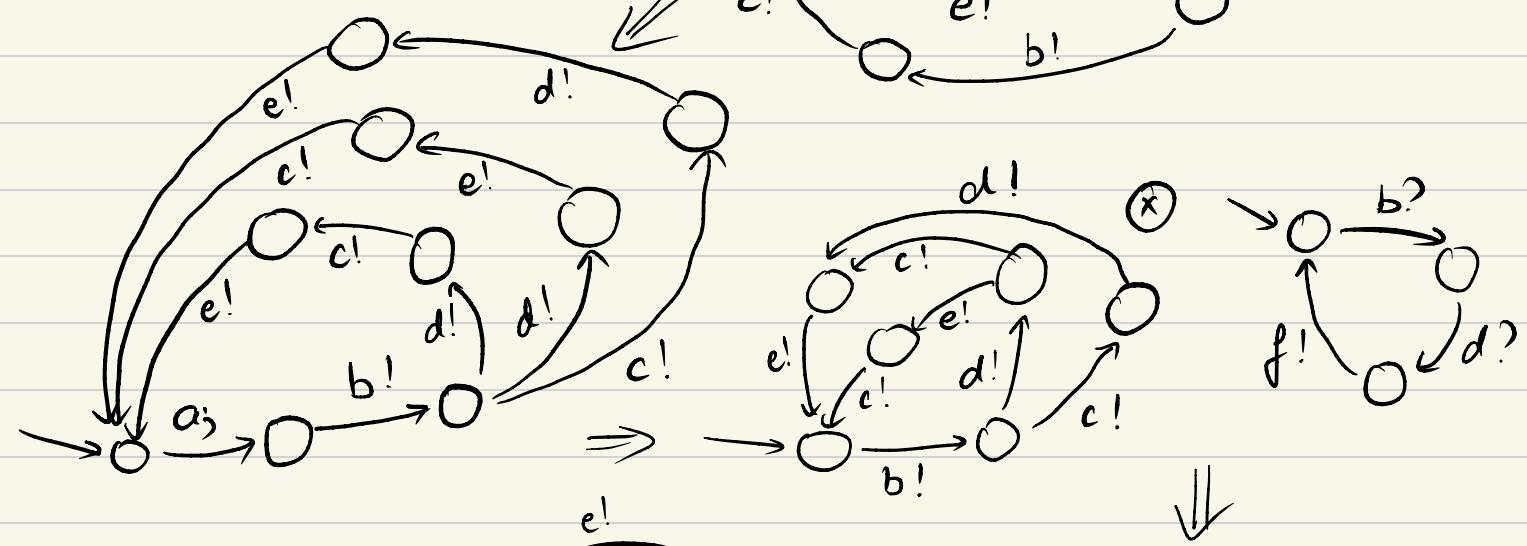
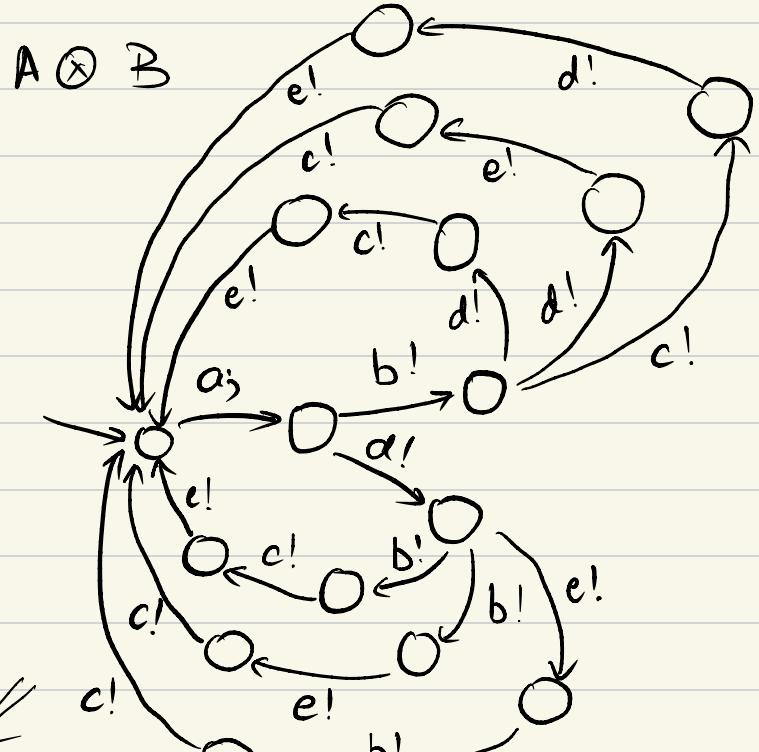
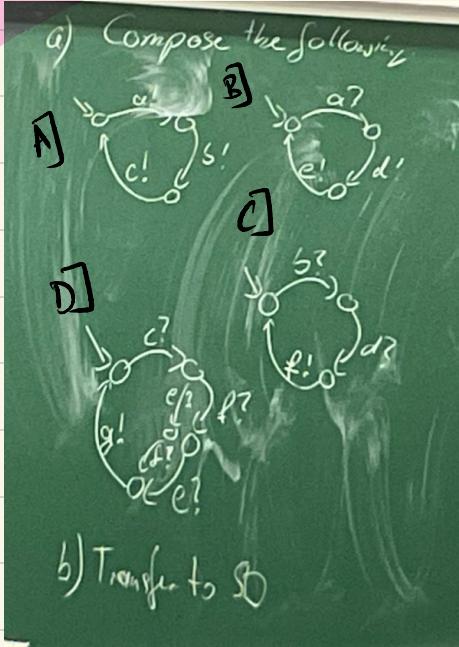


⑤

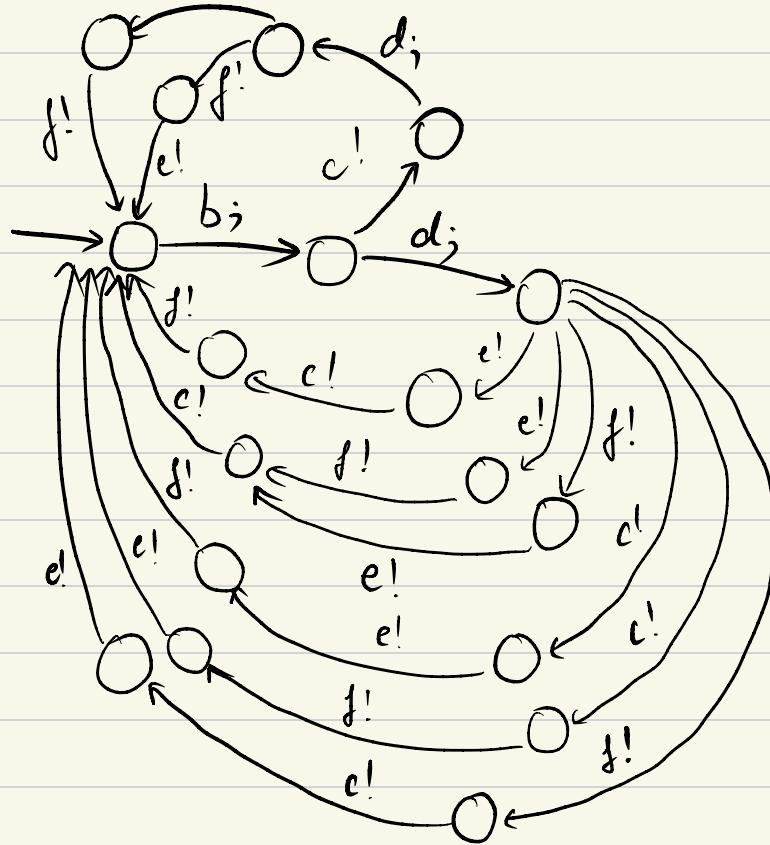


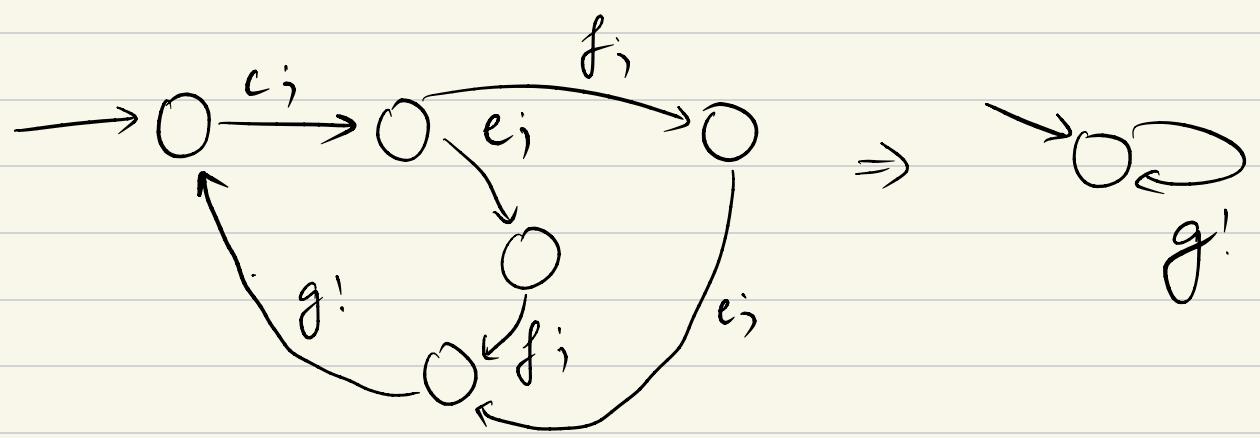
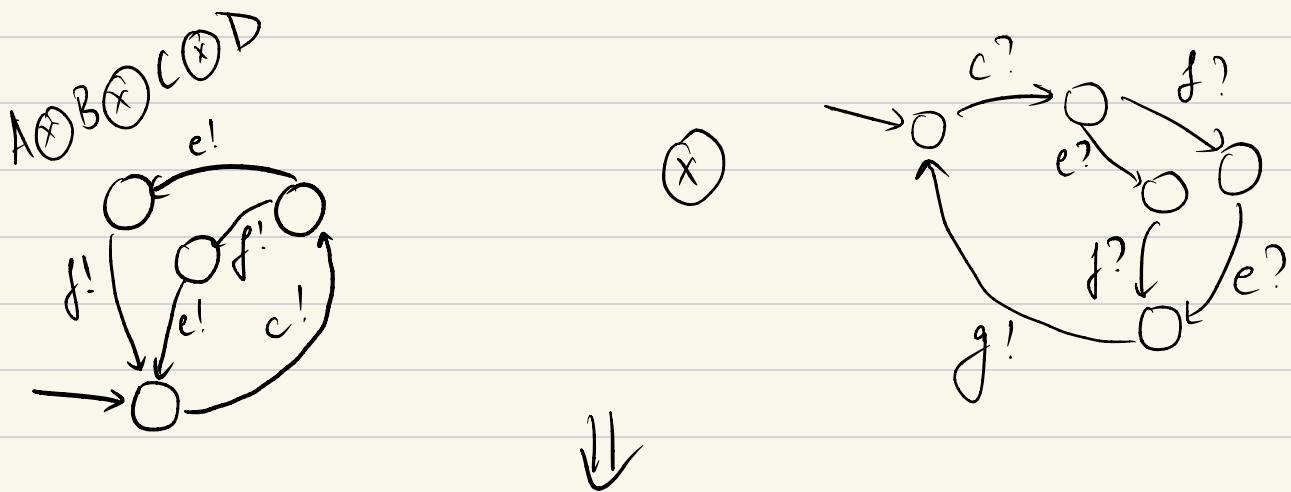
⑥



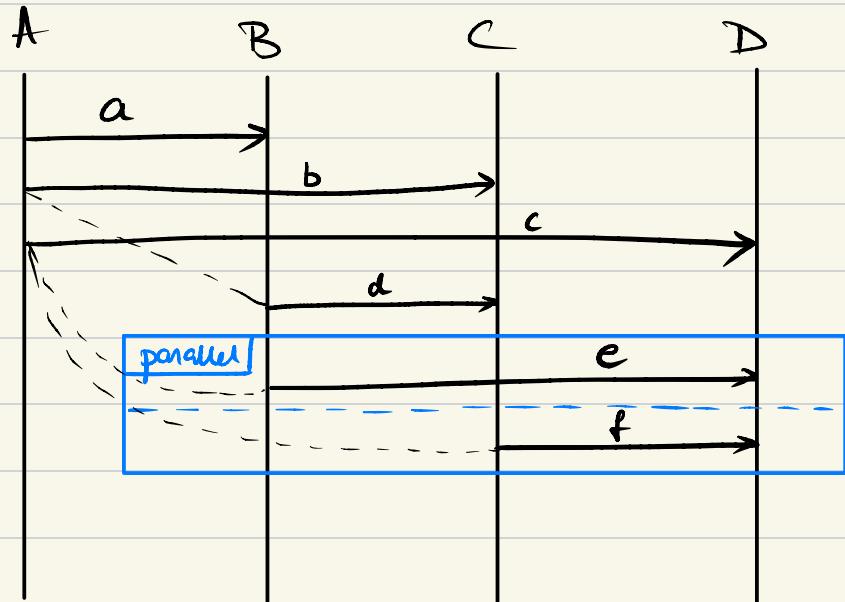


$A \otimes B \otimes C$





(b) transfer to SD



Q

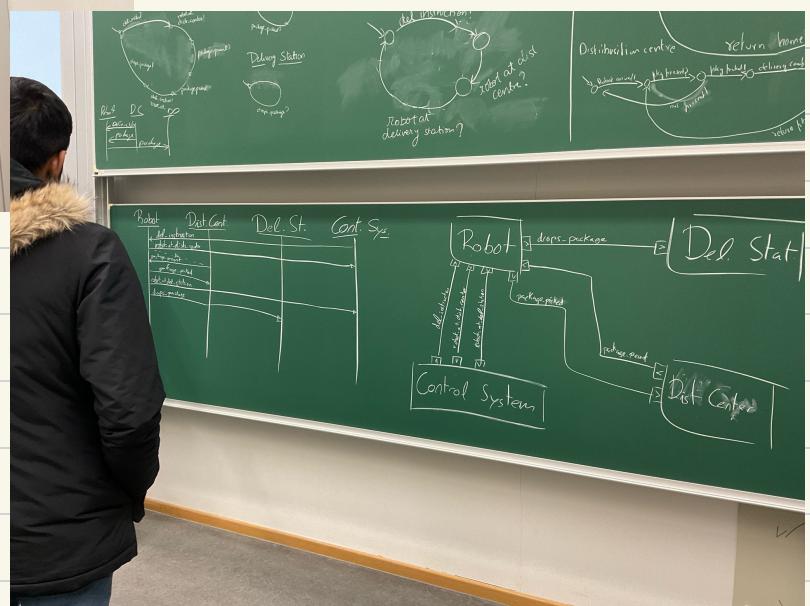
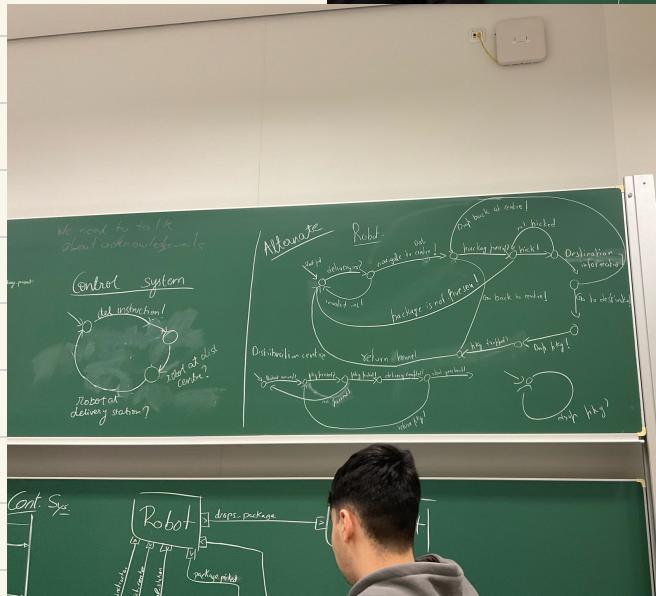
a) Determine Hamming Distance

$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$

→ hamming dist = 5

b) Define the shortest possible code  
with 3 codewords and a  
Hamming Distance of 3

$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix}^3$



Sample  
Exam  
Q.

