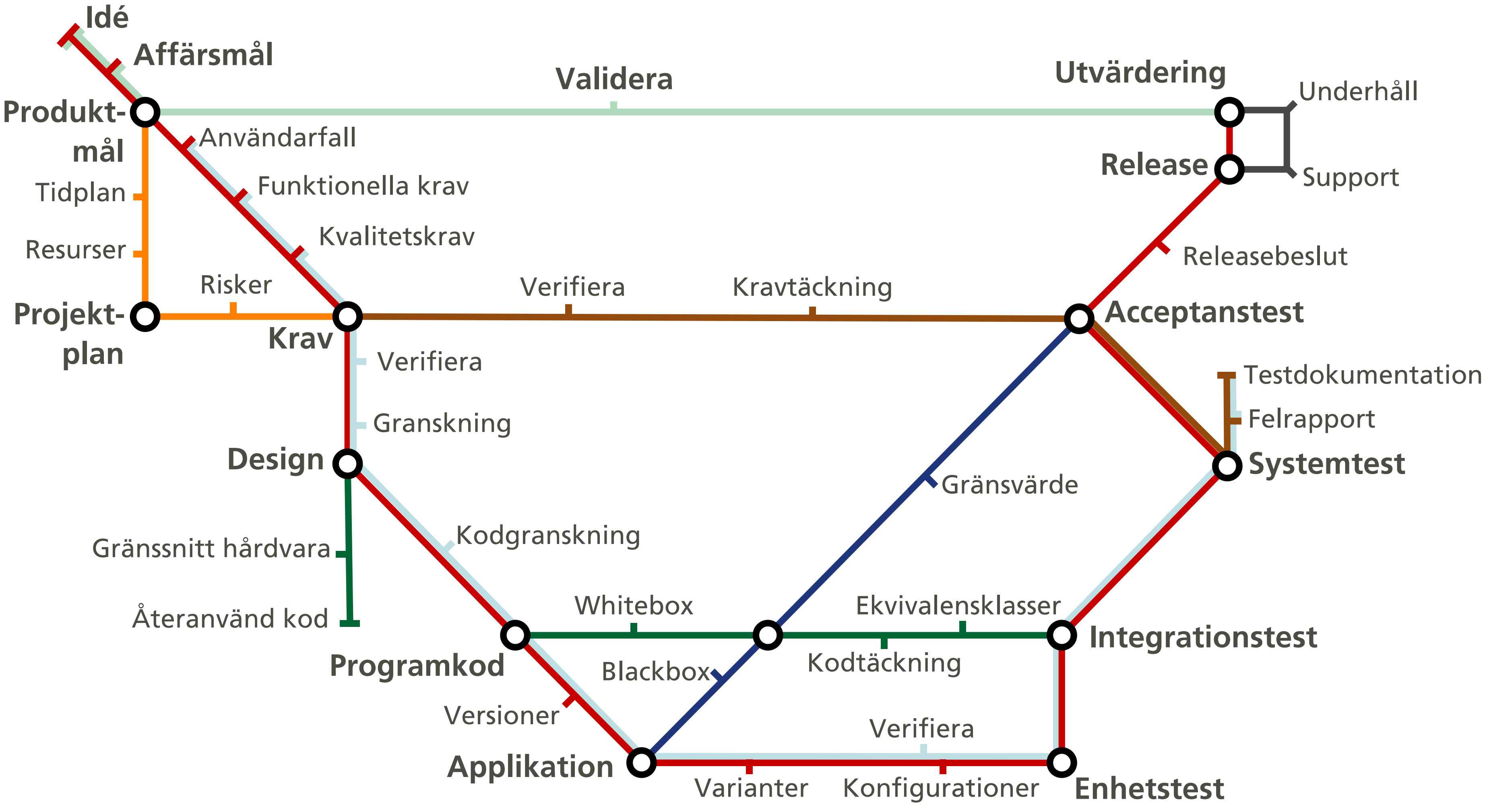
LUND UNIVERSITY

Exercise 1a: Requirements and Project Kick-off

ETSA01 INGENJÖRSPROCESSEN 1 - METODIK VT15


Idé
Affärsmål
Validera
Utvärdering
Underhåll
Produkt-
mål
Användarfall
Release
Support
Tidplan
Funktionella krav
Kvalitetskrav
Releasebeslut
Resurser
Risker
Verifiera
Kravtäckning
Projekt-
plan
Krav
Acceptanstest
Verifiera
Testdokumentation
Felrapport
Granskning
Design
Systemtest
Gränsvärde
Kodgranskning
Gränssnitt hårdvara
Whitebox
Ekvivalensklasser
Återanvänd kod
Integrationstest
Programkod
Blackbox
Kodtäckning
Versioner
Verifiera
Applikation
Varianter
Konfigurationer
Enhetstest

4(+2) Exercise Sessions

Ex 1a: Requirements introduction, project requirements specification kick-off – or “how to enter a garage with a bike”

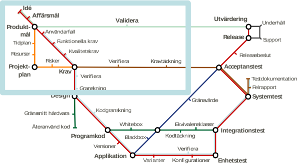
Ex 1b: Requirements evaluation workshop

Ex 2: Project Plan & Inspection

Ex 3: Test I

Ex 4a: Test II, project test specification kick-**on**

Ex 4b: Test case evaluation workshop



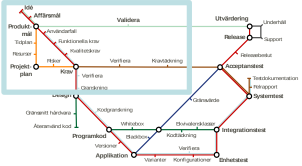
Agenda for Exercise 1a and 1b:

1a Requirements introduction and overview

- Requirements elicitation (ex. R.1 - R.4, R.6)
- ER diagram (ex. R.5)
- Use case
- Project work kick-off

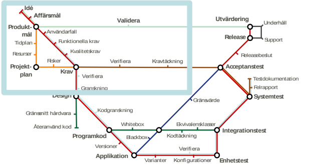
1b Requirements evaluation workshop

- Evaluation:
  - Use case
  - Functional requirements
  - Quality requirements

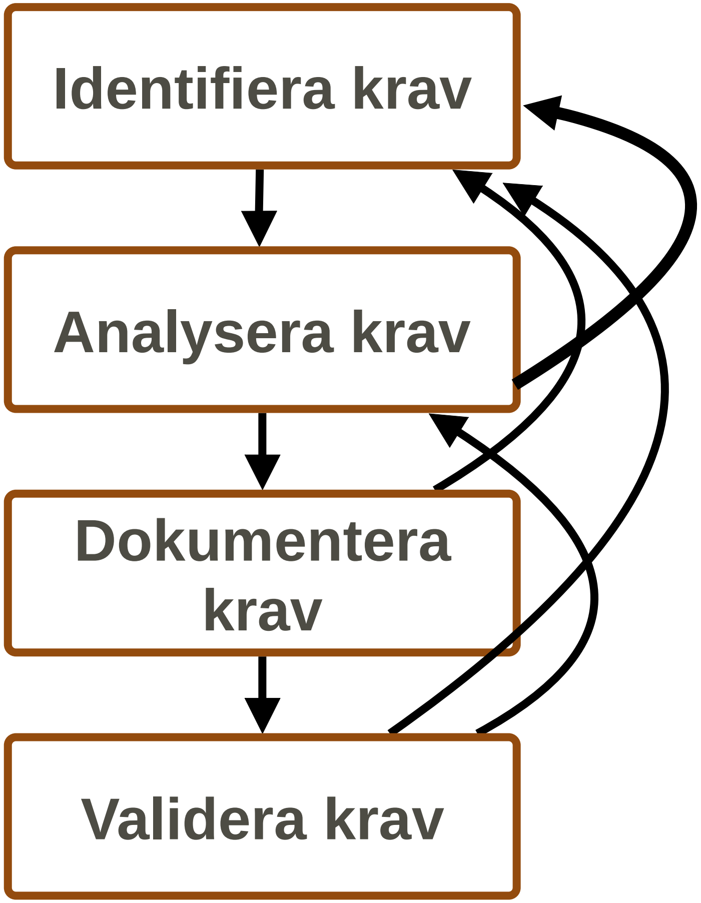


Why and How to Produce a Good SRS?

- SRS – System Requirements Specification
- Importance of the SRS
  - Agreement between client and supplier
  - Basis for implementation of the system
  - Reference for validation of the final product
  - Prerequisite for high-quality software
  - A high-quality SRS reduces the development cost



Phases of Requirements Engineering

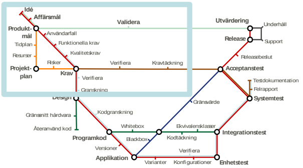


Identifiera krav

Analysera krav

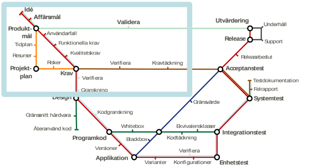
Dokumentera krav

Validera krav



R1: Assignment

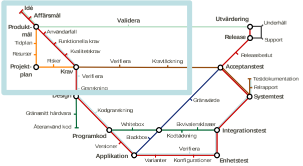
- Identify stakeholders for a course administration system
- Source of requirements for the system
- Possible conflicts between stakeholders' needs/requirements



R1: Assignment

- Students
- Teachers
- Course responsible
- Course secretary
- Study guidance person
- CSN
- System managers – LDC
- University lawyers (PUL)

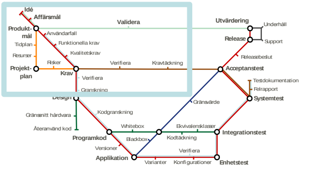
- Security vs. usability
- LDC password policy vs. students’ preferences



R2-3: Quality Criteria for Functional and Quality Requirements

Good requirements should be:

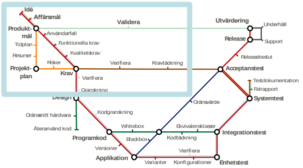
- – Correct
- – Complete
- – Unambiguous
- – Verifiable
- – Consistent
- – Necessary
- – Traceable
- – Ranked for importance



R2-3: Functional vs. Quality Requirements for a Bike Computer

Functional requirements

Quality requirements



R2-3: Functional vs. Quality Requirements for the Bike Computer

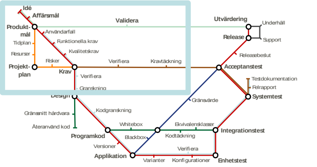
Functional requirements

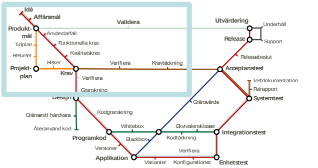
- – The system shall have a function to determine the current speed
- – The system shall have an alarm clock function

Quality requirements

- – 4 out of 5 random cyclist should be able to correctly set the alarm after studying the manual for 5 minutes
- – 95% of the time, the system should respond within 0.1 second after pressing a button
- – The distance measure may diverge 0.5%

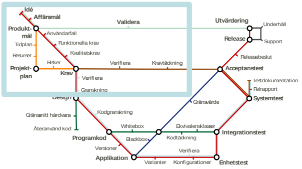
R4: Why Should Requirements be Expressed on Different Abstraction Levels?



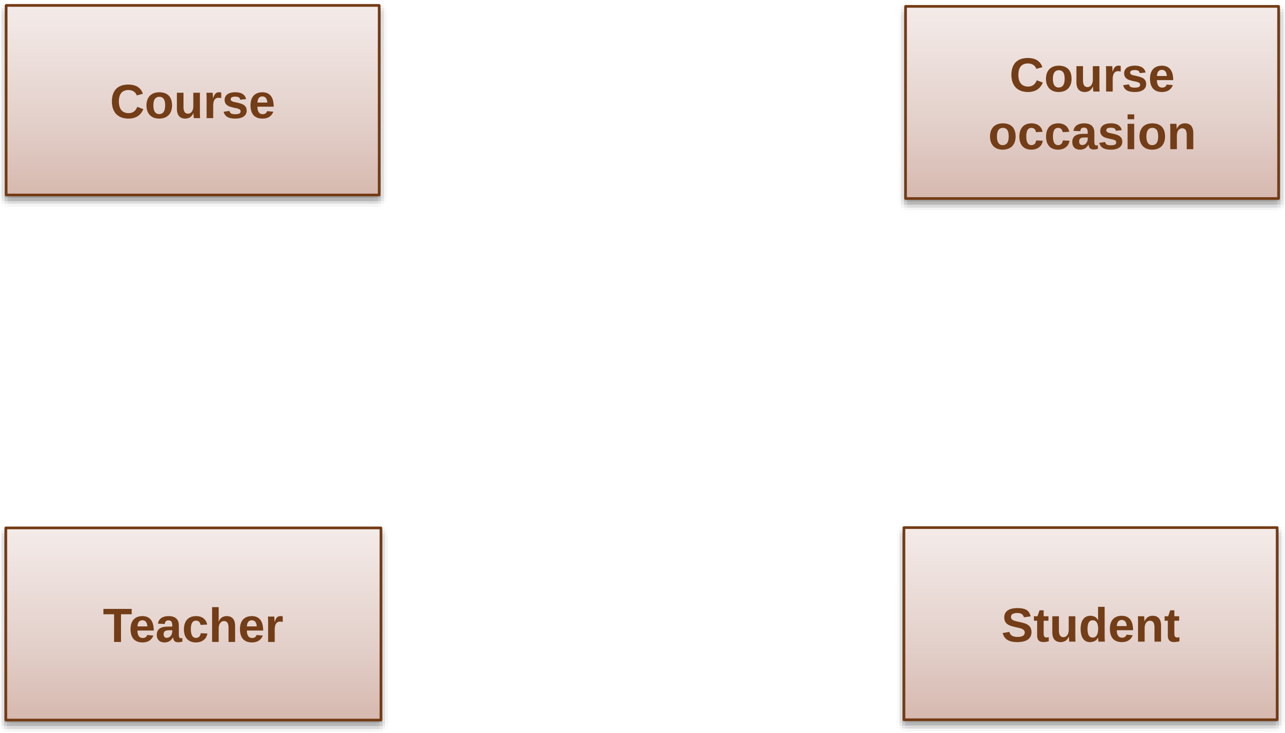


R4: Why Should Requirements be Expressed on Different Abstraction Levels?

- Various types of stakeholders understand different levels of abstractions
- Different purposes for different abstraction levels
- Corresponds with the different steps of writing requirements, from high-level to more detailed
- To build a consistent document structure



R5: ER Diagram for Course Administration System

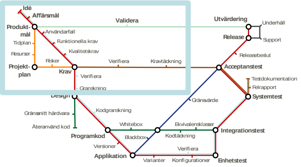


Course

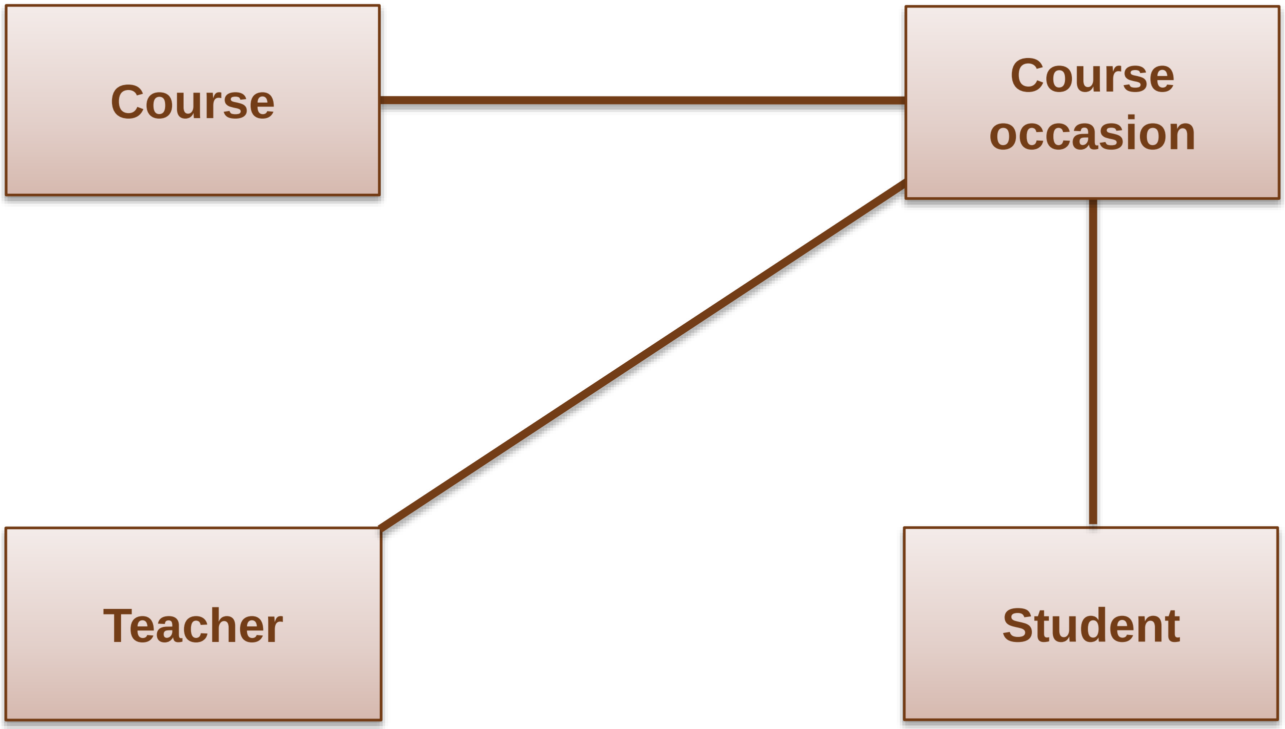
Course occasion

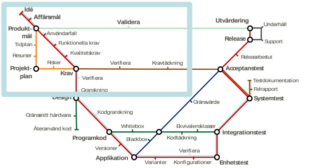
Teacher

Student

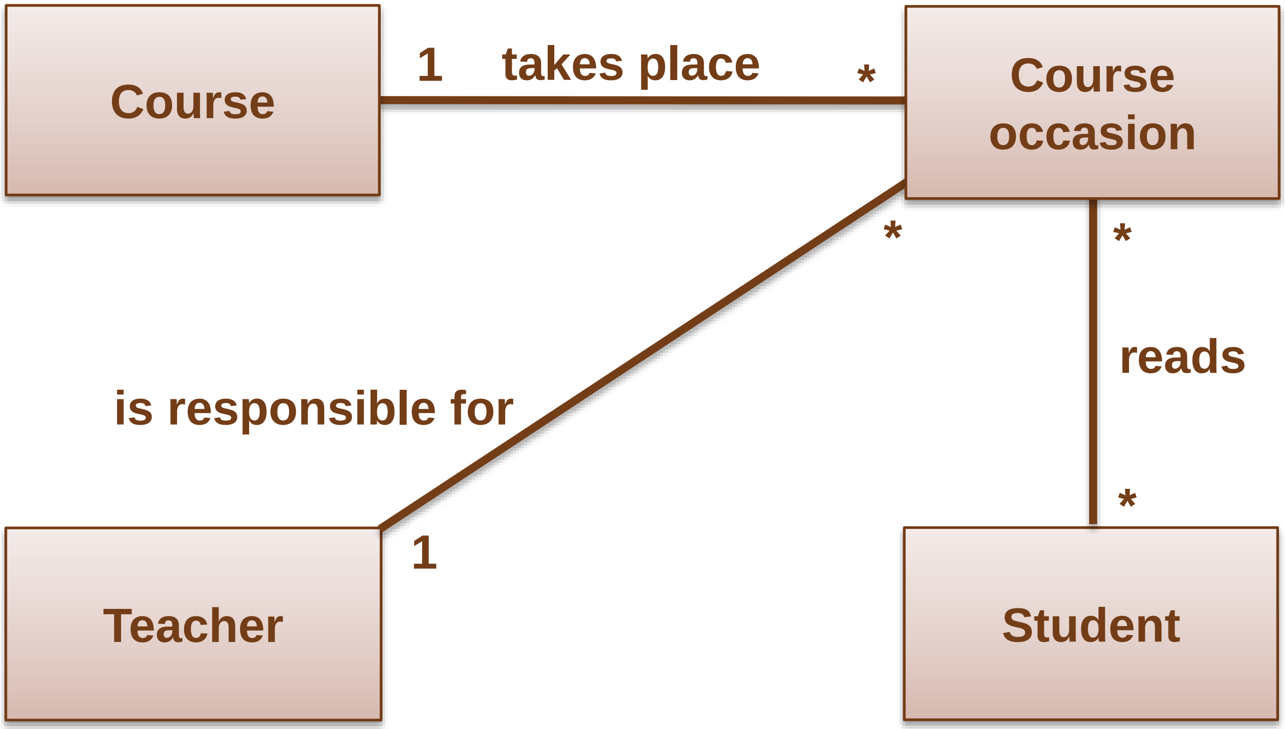


R5: ER Diagram for Course Administration System





R5: ER Diagram for Course Administration System



Course

1

takes place

*

Course occasion

*

*

reads

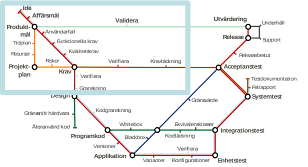
is responsible for

*

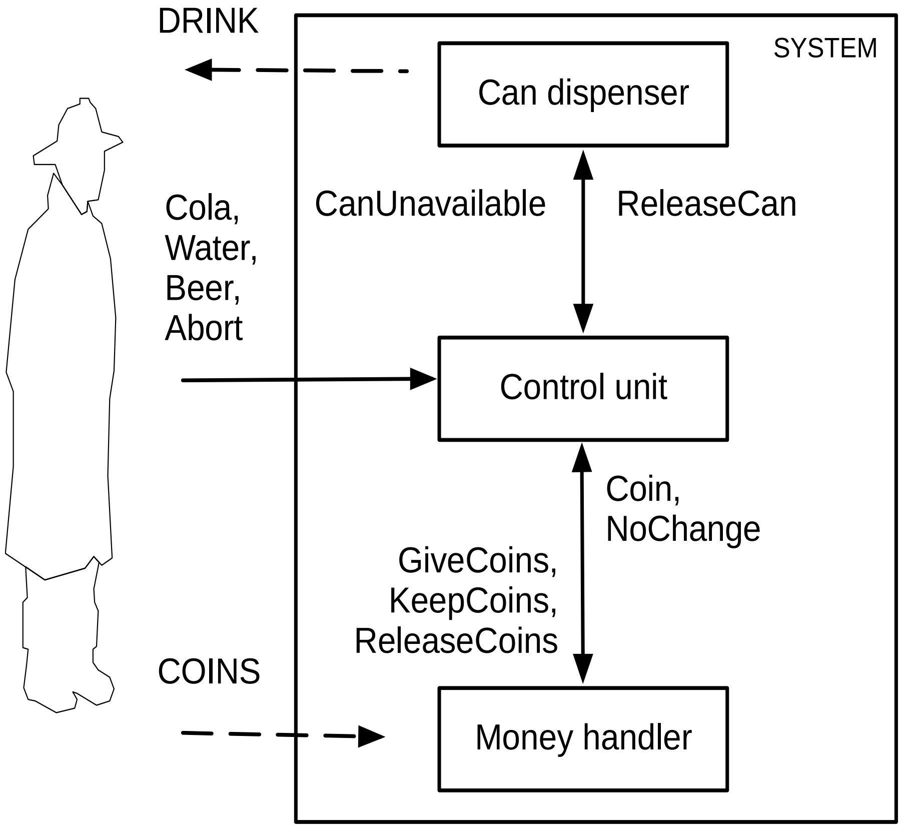
1

Teacher

Student

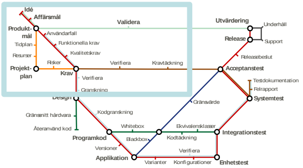


R6: Add Maintenance to the Can Machine



- Interaction requirements describe the interactions between two functions of the system = combinations
- It often also collects all ‘strange’ cases (e.g. unlikely exception scenarios)

R6:
Add maintenance to the can machine



New requirement:

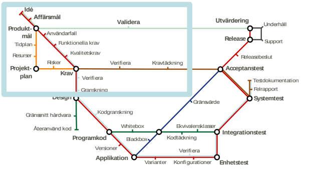
- The machine shall have a maintenance mode during which no drinks can be dispensed

Dependencies among requirements:

- All inserted money shall be returned when the machine is put in maintenance mode
- Money inserted during maintenance mode shall immediately be returned
- Pressed buttons during maintenance mode shall be ignored

Elicitation techniques:

- Matrix with functions both horizontally and vertically and try to find out how they interact
  - Functions can sometimes also interfere with themselves!



Example Use Case for a Can Machine

UC1: User is buying a beer

Primary Actor: User

Precondition: User has enough money to buy a beer, can machine is ready

Main success scenario:

1. User presses the beer button
2. User inserts 2 times 10 SEK and 2 times 1 SEK coins
3. The beer can is released to the user

Exception scenarios:

1a. Machine is out of beer cans

* System displays a message “Out of cans”

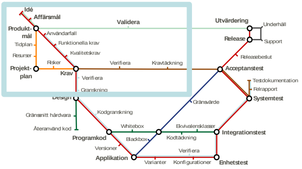
* System returns all coins as soon as they are inserted

3a. User has inserted too much money

* System displays message “Change released”

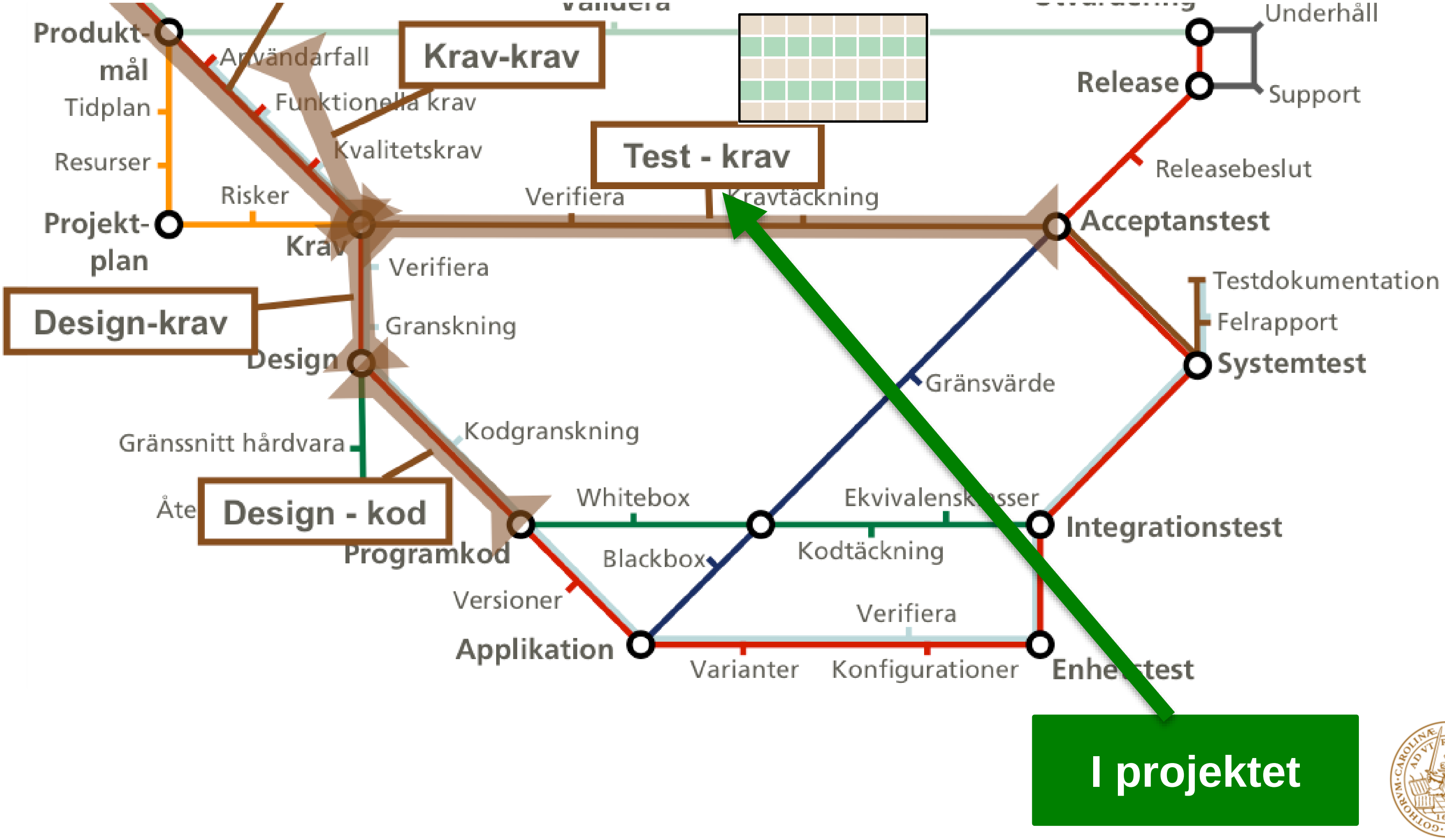
* System calculates the change and releases it through the coins return

Requirements Evaluation



Bike computer		Correct	Complete	Unambiguous	Consistent	Verifiable	Necessary	Traceable	Ranked
F1	The system shall have a function to determine the current speed	✓		✓	✓	✓	✓	?	-
F2	The system shall have an alarm clock function			?	✓	✓	-	?	-
Q1	4 out of 5 random cyclist should be able to correctly set the alarm after studying the manual for 5 minutes			✓	✓	✓	⅔	?	-
Q2	95% of the time, the system should respond within 0.1 second after pressing a button			✓	✓	✓	⅔	?	-
Q3	The distance measure may diverge 0,5%			✓	✓	✓	✓	?	-

Traceability (Spårbarhet)

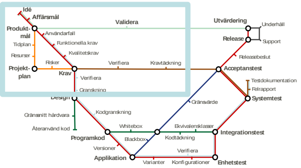


Beställning: Utveckla Användarfall: Ta in cykel i garaget - beställning

1. En cykelägare kommer med en cykel till garagets ingång.
2. Cykelägaren läser cykelns streckkod m h a streckkodsläsare vid ingången.
3. Ingångsdörrens lås öppnas.
4. Cykelägaren placerar sin cykel i garaget och låser sin cykel.
5. Cykelägaren lämnar garaget genom extrautgången.

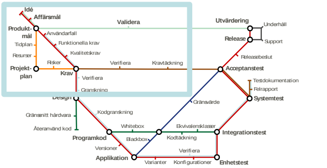
Korrekt	
Heltäckande	?
Otvetydigt	✓
Konsistent	?
Verifierbart	?
Nödvändigt	?
Spårbart	?
Rankat	

Project Work Before Requirements Evaluation Workshop



Now: Plan the week and W2

1. Set up project web
   - Test user permissions
   - Start SRS 0.1
2. Create your Use Case 1: Bike owner enters the garage
   - Actors
   - Pre-conditions
   - Success scenario
   - Exception scenarios
3. Related to Use Case 1:
   - Write functional requirements
   - Write quality requirements
   - Discuss business and product goals
   - Evaluate and redo
4. Assign responsible for the SRS 0.1?
5. Bring print-outs for all project members to **Requirements evaluation workshop Ö1b (tomorrow)**



Hints

For structuring the use case, functional requirements and quality requirements

- Discuss business and product goals
- Study Jalote, Ch 4
- Study course compendium 2.5.2
- Study example in these handouts
- Study example in lecture handouts

Alltså: Uppdrag till Ö1b

- **Innan lokalen lämnas:** detaljplanera vem-vad-när-var → Ö1b
- Läs in projektbeskrivningen avsnitt 1-4. Fokus vad vill/hur tänker beställaren
- Få igång projektwebben
- Utveckla användarfallet i projektwebben Krav 0.1
  - Utveckla beteendet
  - Formulera undantag och förslå respons från systemet
- Formulera relaterade och förtydligande **funktionella krav** och **kvalitetskrav**
- Projekthandledarna tittar in i projektwebben någon timme innan Ö1b
- Ta med utskrifter av användarfall, skall-krav och kvalitetskrav till övningen