LUNDS UNIVERSITET

Föreläsning 3: Test II, Design II, Robotmässa

Programvaruutveckling - Metodik 2018 | Markus Borg

Kursombud

2 X C:

Olivia Mattsson
- ol3270ma-s@student...

Ellen Widing
- el7116wi-s@student...

2 X I:

TBD

TBD

Var är I?

LUNDS UNIVERSITET

Agenda F3

Programvarutestning - del 2

- Black-box testning

Programvarudesign - del 2

- Arkitekturdesign

Robotmässa

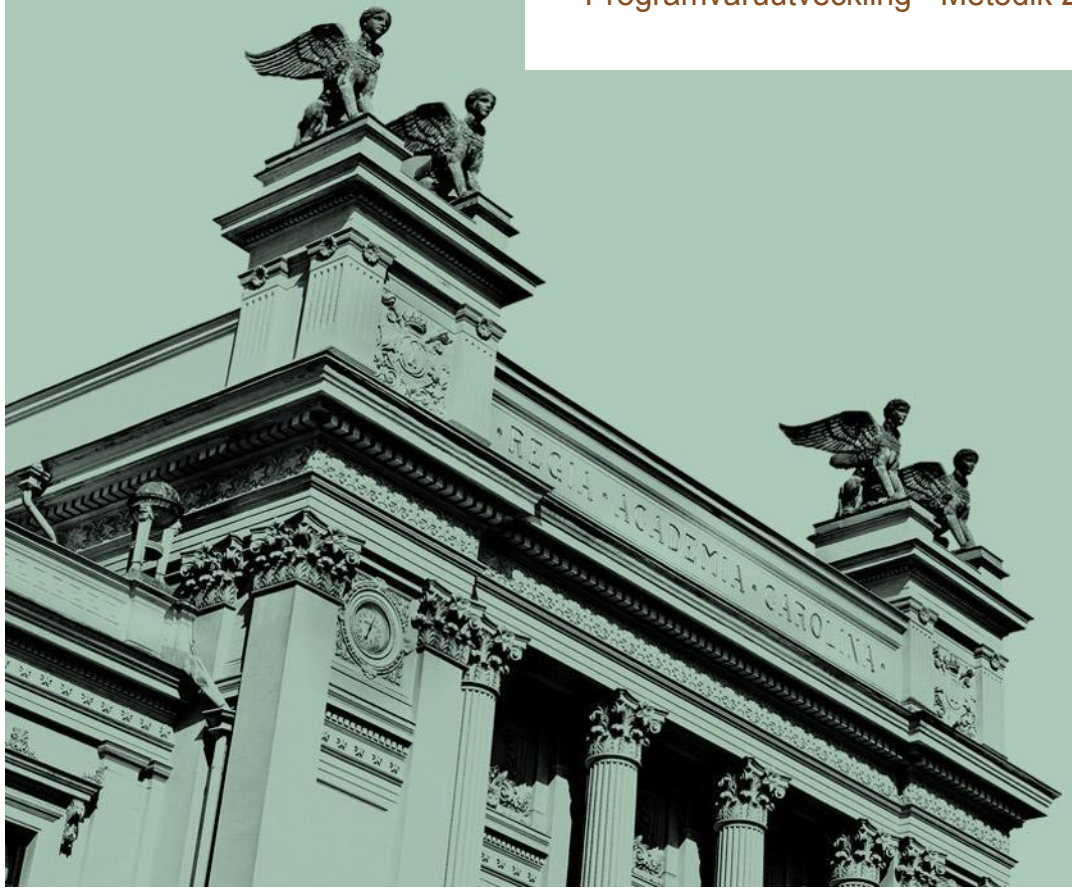
- Status i projekten
- Videovisning

LUNDS UNIVERSITET

LUNDS UNIVERSITET

Test II

Programvaruutveckling - Metodik 2018 | Markus Borg



Black-box vs. White-box

White-box

- Kräver tillgång till koden
- Testar utfall och inre funktion
  - täcker vi raderna?
  - täcker vi vägarna?

Black-box

- Programmet ses som en ”svart låda” och man utnyttjar inte någon kunskap om koden i samband med definition av testfall
- Kravspecifikationen används för att ta fram testfall
- Testar utfall/resultat

LUNDS UNIVERSITET

Black-box-testning

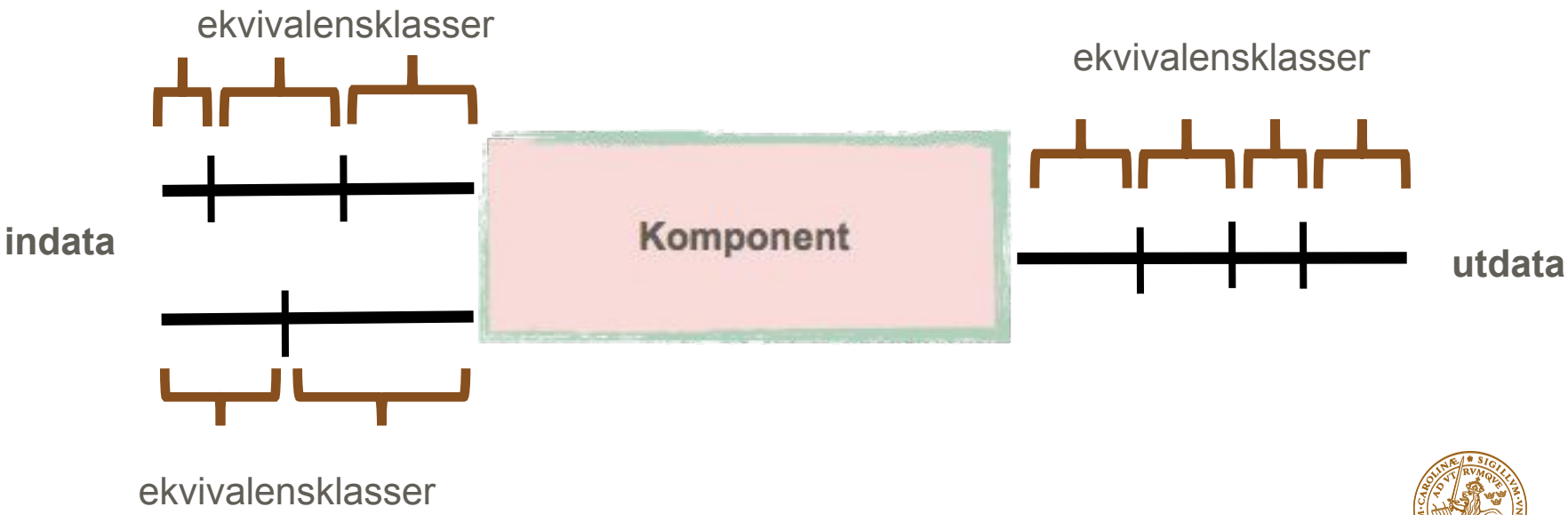
Några vanliga tekniker för testdesign

- Ekvivalenspartitionering
- Gränsvärdestestning
- Stresstestning
- Parvis testning

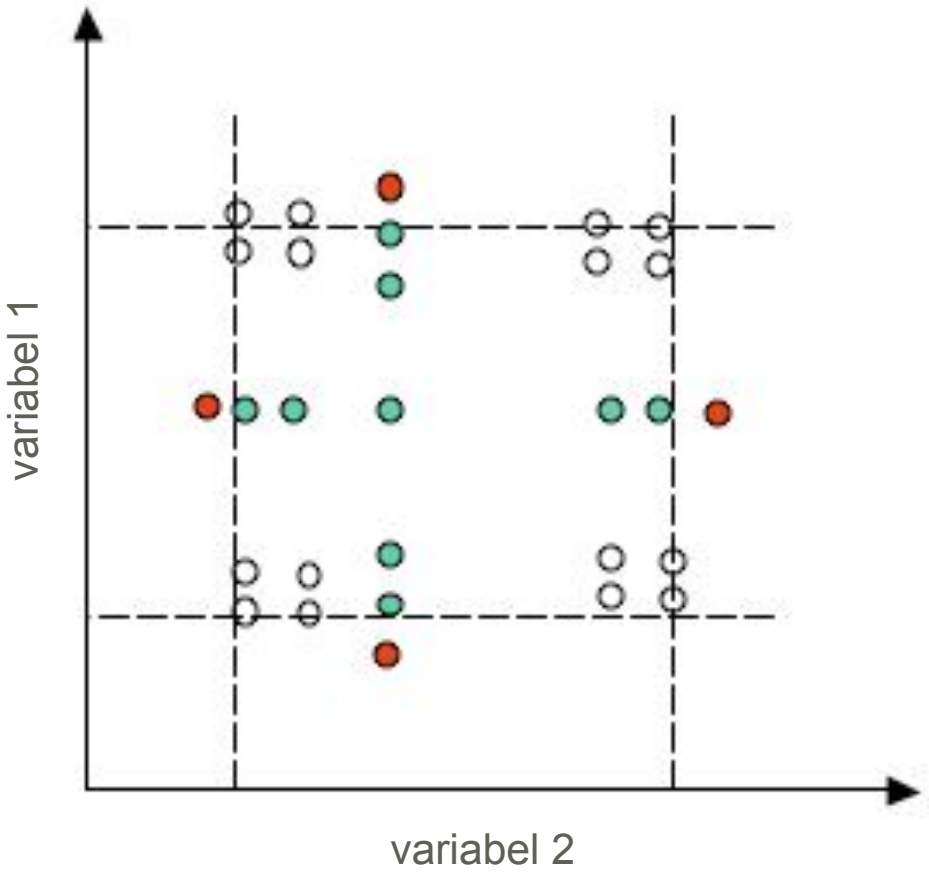
LUNDS UNIVERSITET

Ekvivalenspartitionering

Hitta värden för in och utdata som behandlas på inbördes enhetligt sätt



Gränsvärdestestning
- exempel med två variabler



För n variabler, ett intervall var

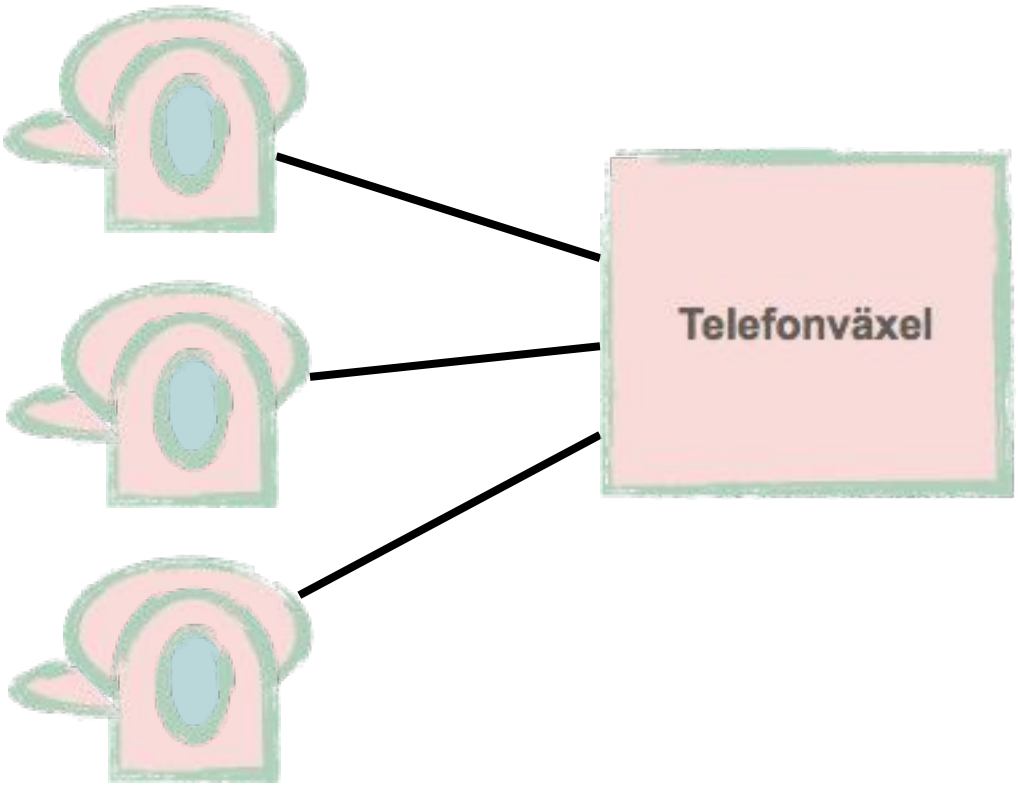
- Vanliga gränsvärden: innanför eller på gränser => 4n+1 testfall

- Robust-test: även utanför gränserna => 6n+1 testfall

- Kritisk systemtestning: även alla kombinationer av gränsfall: 5^n testfall

Stresstestning

Kontrollera vad som händer vid hög belastning, t ex:



telefoner > max
samtidiga samtal > max
…

ERROR: ArrayOutOfBounds ?

Parvis testning

En strategi för testdesign som reducerar antalet testfall

- Test bör undersöka kombinationer av parametrar
- Antalet möjliga testfall blir fort väldigt stort
  - » 5 olika operativsystem
  - » 10 olika versioner av Java
  - » 15 olika webläsare
  - » …
- Istället för att testa alla kombinationer, testa alla parameterpar
  - » Visat sig vara en effektiv strategi (dvs. hittar stor andel buggar)

5 x 10 x 15 = 750 testfall

LUNDS UNIVERSITET

Parvis testning för systemtest av diskmaskin

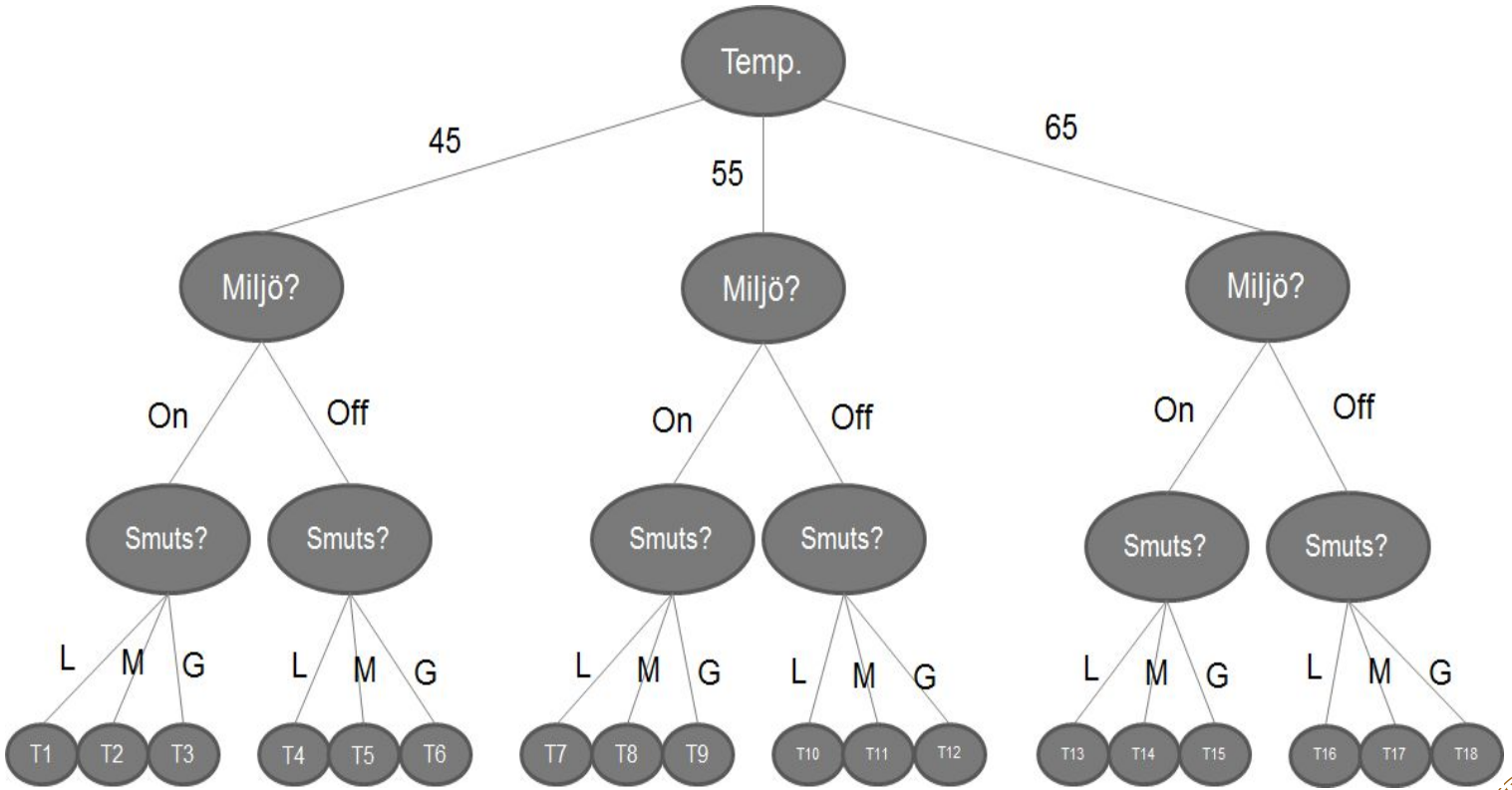
Testparametrar:

Temperatur	(45, 55, 65)
Miljöläge	(on, off)
Nedsmutsningsgrad	(lätt, måttlig, grov)

Utfall:

Diskresultat	(rent, smutsigt)

Samtliga möjliga testfall



⇒ **3 x 2 x 3 = 18 möjliga testfall**

LUNDS UNIVERSITET

Använd parvis testning!

Samtliga kombinationer av parameterpar testas

Färre testfall än vad som krävs för uttömmande testning

- Men en rimlig nivå för att hitta defekter

Generera testdata som uppnår parvis täckning med ett minimalt antal testfall är svårt

- ett kombinatoriskt optimeringsproblem
- verktyg används för att generera testdata

LUNDS UNIVERSITET

Systemtest diskmaskin: Möjliga par

Temperatur-Miljöläge (3 x 2 komb.)

- 45-on, 45-off
- 55-on, 55-off
- 65-on, 65-off

Temperatur-Nedsmutsningsgrad (3 x 3 komb.)

- …

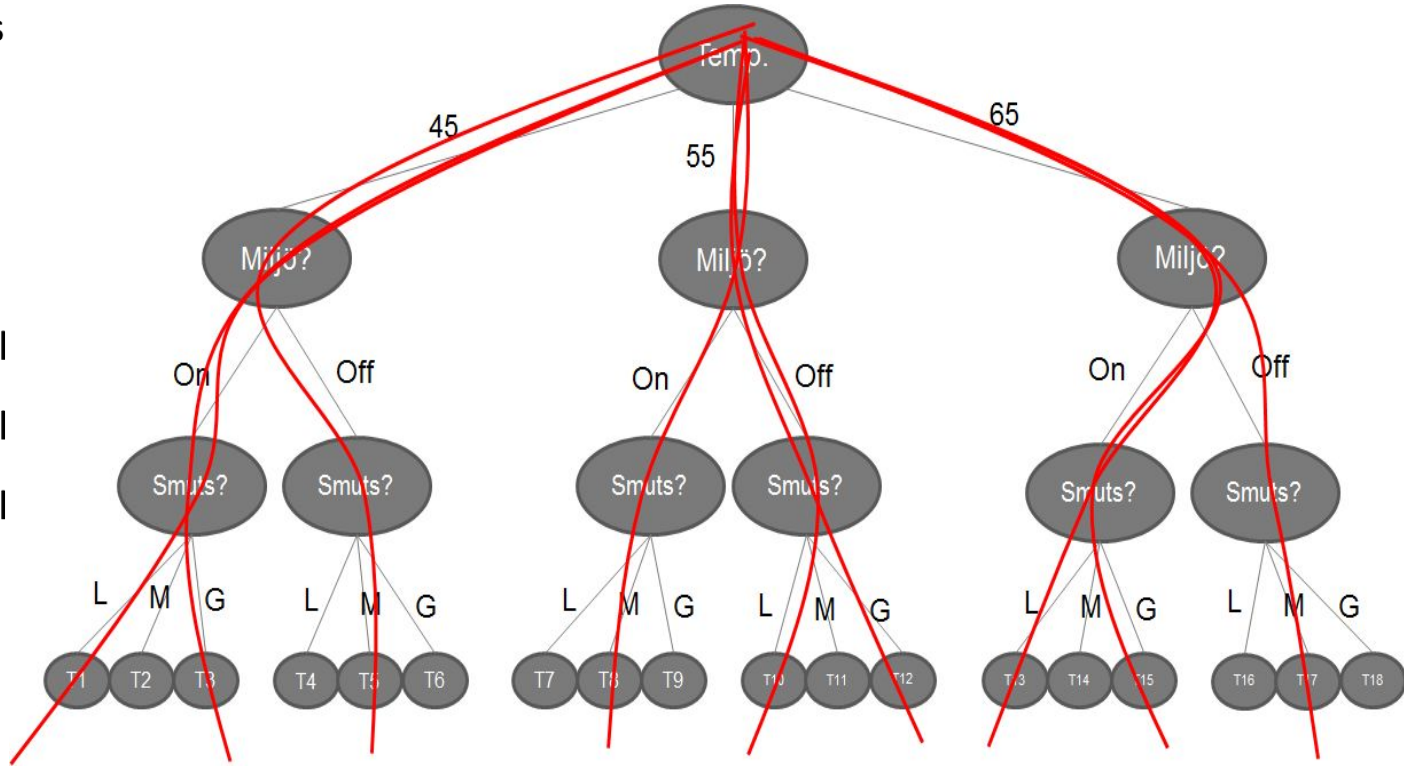
Miljöläge-Nedsmutsningsgrad (2 x 3 komb.)

- …

LUNDS UNIVERSITET

Lösning med 9 testfall => halverat testbehov

	Temp.	Miljö.	Smuts
T1	45	On	Lätt
T2	55	Off	Lätt
T3	65	On	Lätt
T4	45	Off	Medel
T5	55	On	Medel
T6	65	Off	Medel
T7	45	On	Grov
T8	55	Off	Grov
T9	65	On	Grov



→ **9 testfall räcker för att testa alla parameterpar**

Genomföra, dokumentera och rapportera test

I simulerad miljö

- endast i utvecklingsmiljön (SW)

I mer verklig miljö

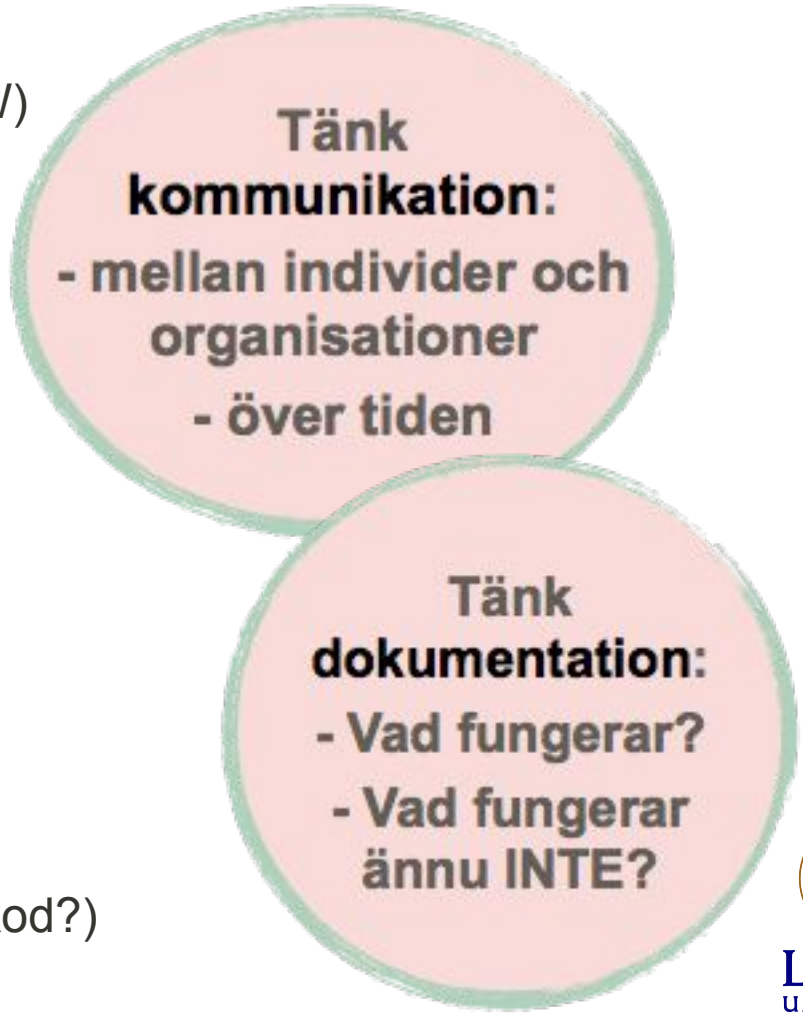
- i testuppsättning (SW/HW)

I verklig miljö

- i äkta (SW/HW)
  med äkta användare

Alla fel rapporteras/registreras

- Testfall
- Resultat
- Feltyp
- Allvarlighet
- Ev ytterligare beskrivning (felkod?)



LUNDS UNIVERSITET

Slutord - Test

Testning ofta den enskilt dyraste aktiviteten i ett utvecklingsprojekt

När är testningen färdig?

Finns inga garantier att alla defekter är hittade!

- Man kan alltid testa mer…

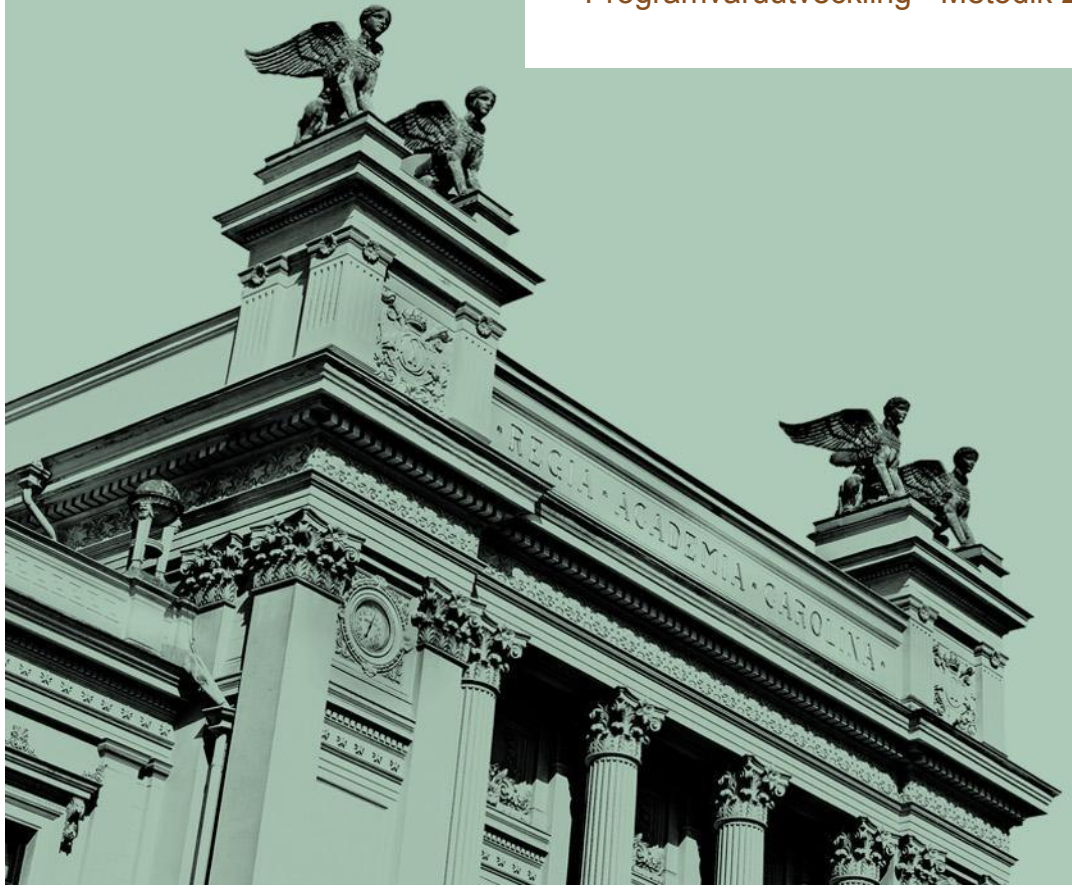
- Slösade resurser
- Försenad release
- Ökade kostnader

- Defekter finns kvar
- Missnöjda användare
- Dyr support
- Dyrt underhåll

LUNDS UNIVERSITET

Programvarudesign 2

Programvaruutveckling - Metodik 2018 | Markus Borg



Programvarudesign - agenda

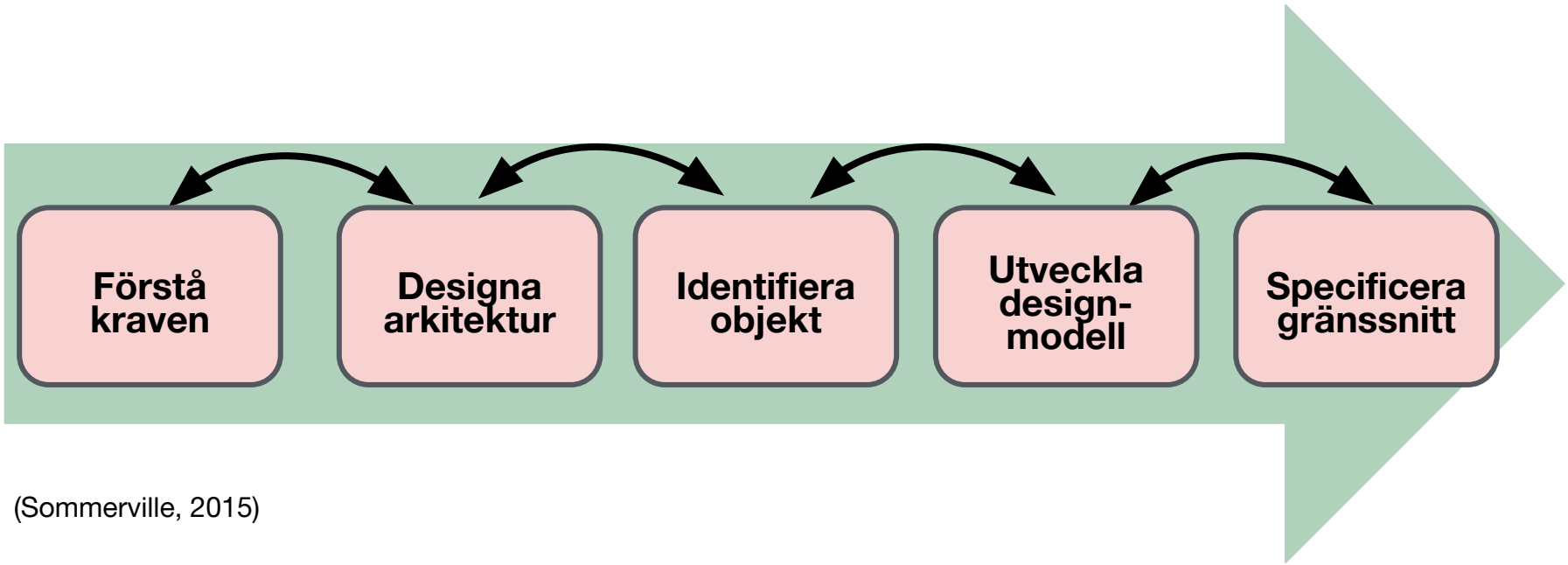
- F2: Objektorienterad design
- F3: Arkitekturdesign
- (Ej: Interaktionsdesign)

Objektorienterad design

- Implementationsnära design
- Beskrivning av hur komponenter implementeras på klassnivå
- Klasser beskriver meningsfulla entiteter i problemdomänen
  - Substantiv i beskrivningen blir klasser
  - Operationer implementeras i metoder

LUNDS UNIVERSITET

Modell för design av objektorienterad programvara

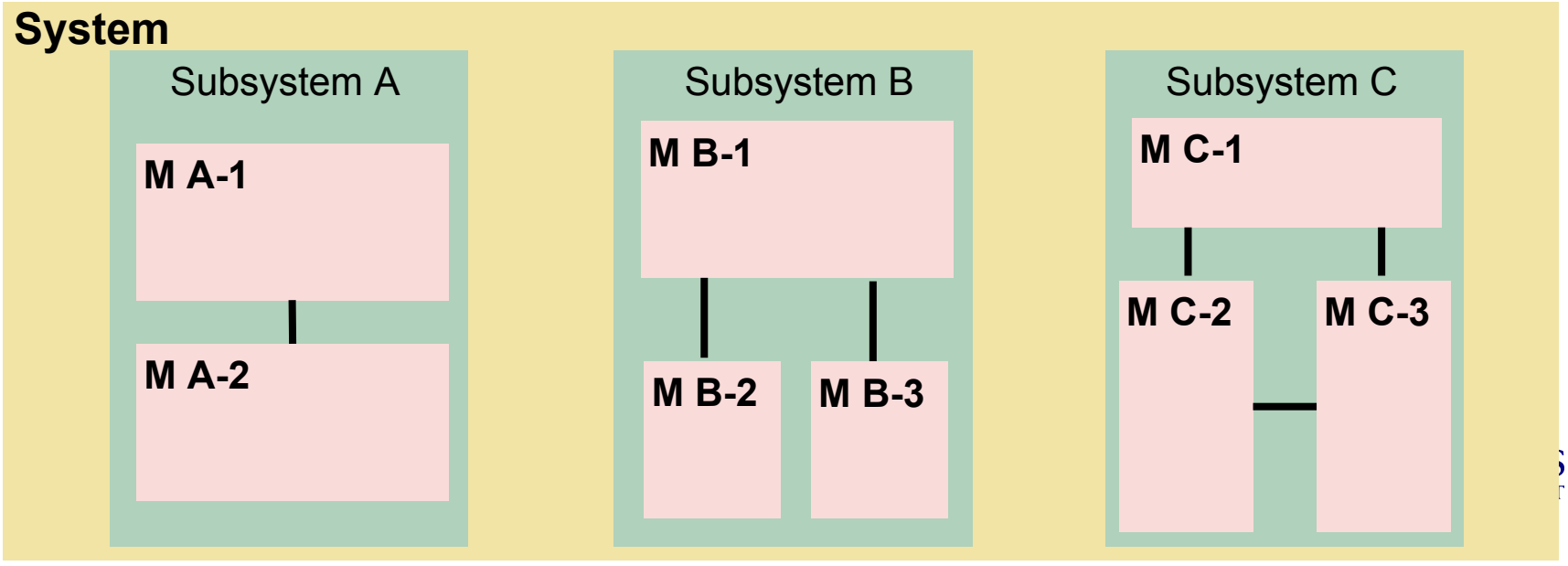


(Sommerville, 2015)

Arkitekturdesign

Nedbrytning av systemets övergripande struktur

- System - helheten
  - ↪Subsystem – enhet som ej beror på andra subsystem
    - ↪Moduler – enhet som verkar ihop med andra moduler
      - ↪(Komponenter – en eller flera klasser)



system = samling komponenter som samverkar för att uppnå ett mål

system-av-system = samling system som samverkar för att uppnå mål utöver summan av de ingående systemen (framträdande egenskaper)

Från militära tillämpningar till civilt bruk (t.ex. trafikmiljö eller industri 4.0)

Syfte med arkitekturdesignen

- Länk mellan kraven och detaljerad design
  - Grov ritning för implementation
- Kommunicerar designbeslut i organisationen
- Grund för systemanalys
  - Säkerhet
  - Prestanda
- Underlättar återanvändning
  - Använda delar i andra system
  - Utveckla produktlinjer

LUND UNIVERSITY

Val av arkitekturdesign

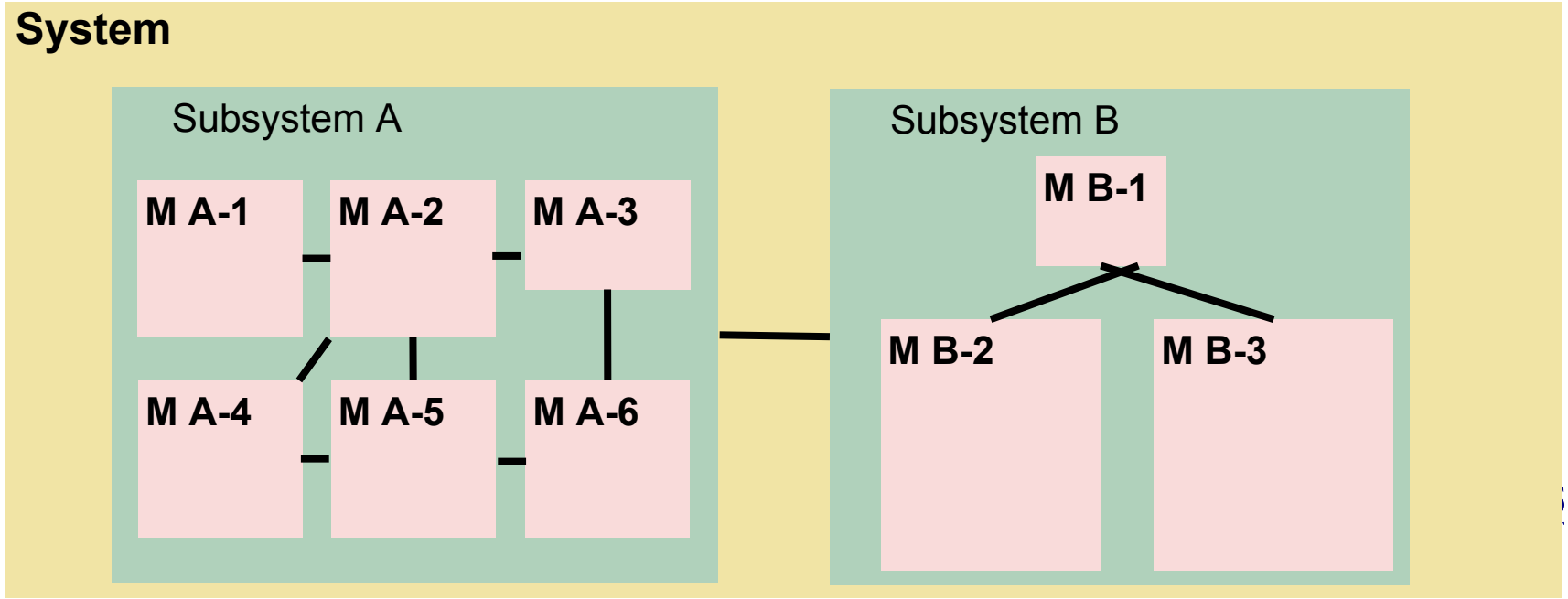
- Förståelse för kontext och intressenter nödvändig för bra beslut
- Kvalitetskraven avgör ofta beslutet
  - Arkitekturellt signifikanta krav
- Vad vill vi uppnå? Motsättningar vanligt!

Övriga faktorer som kan avgöra

- Organisationens tekniska kompetens och erfarenhet
- Återanvändning av tidigare arkitektur
- Standarder som behöver uppfyllas

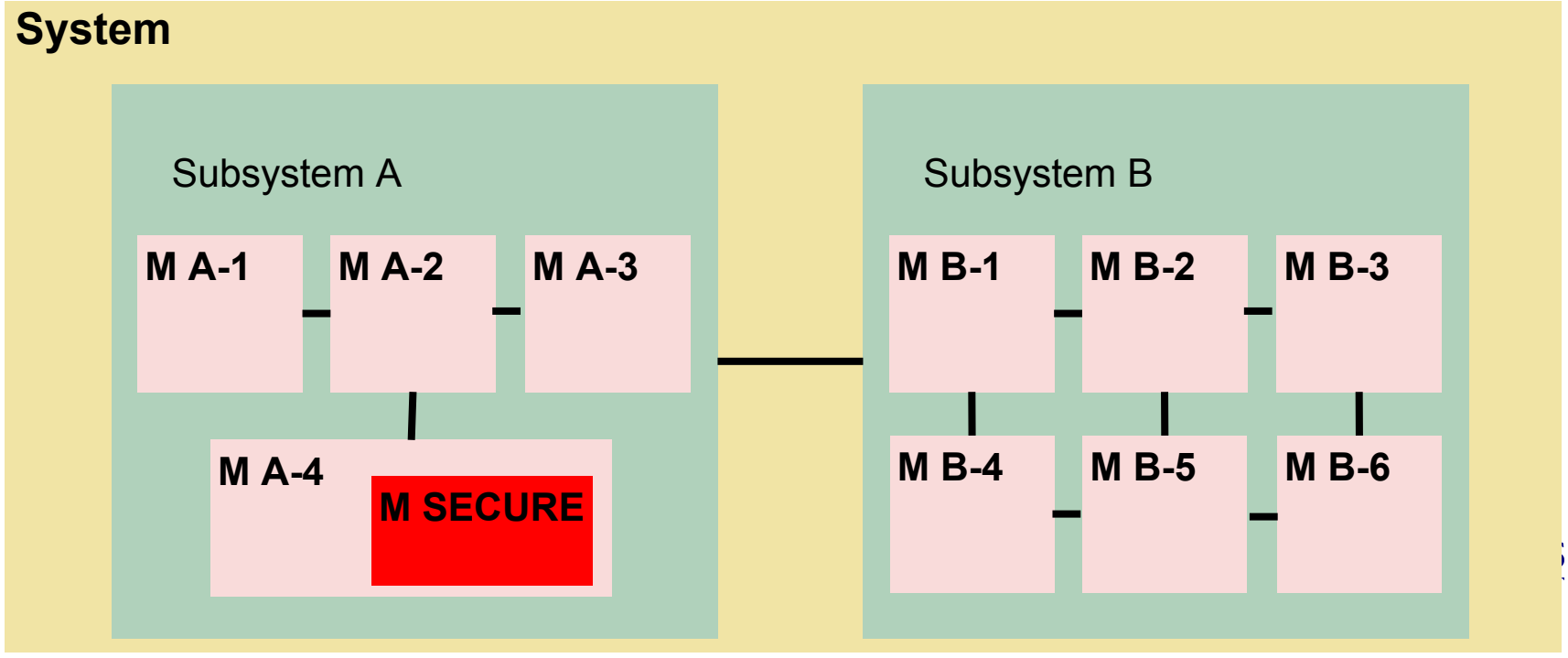
Prestanda?

- Kommunikation är en prestandatjuv!
- Samla tunga beräkningar i moduler som kommunicerar minimalt utåt
- Acceptera att beräkningsmoduler blir stora



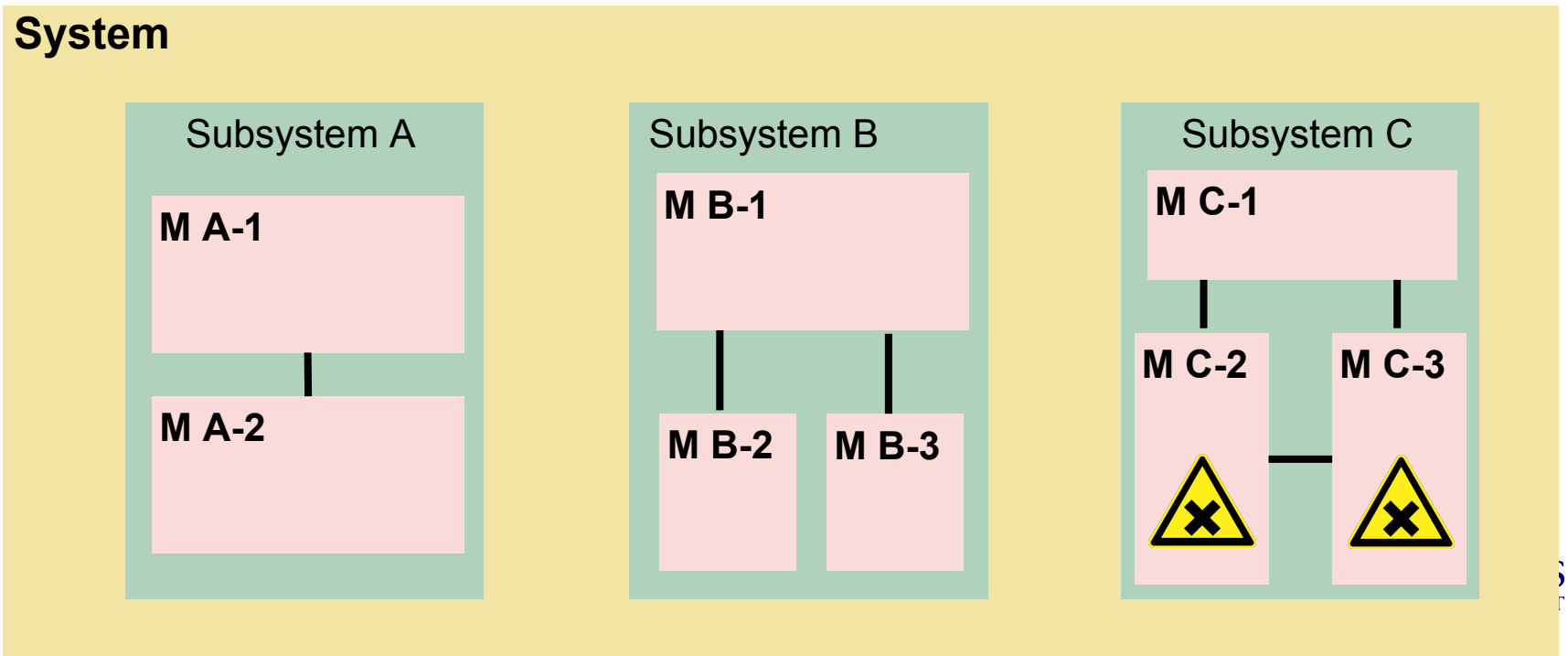
Säkerhet? (security)

- Åtkomstbegränsning viktigt
- Introducera säkerhet i olika lager
- Hantera den känsligaste informationen innerst



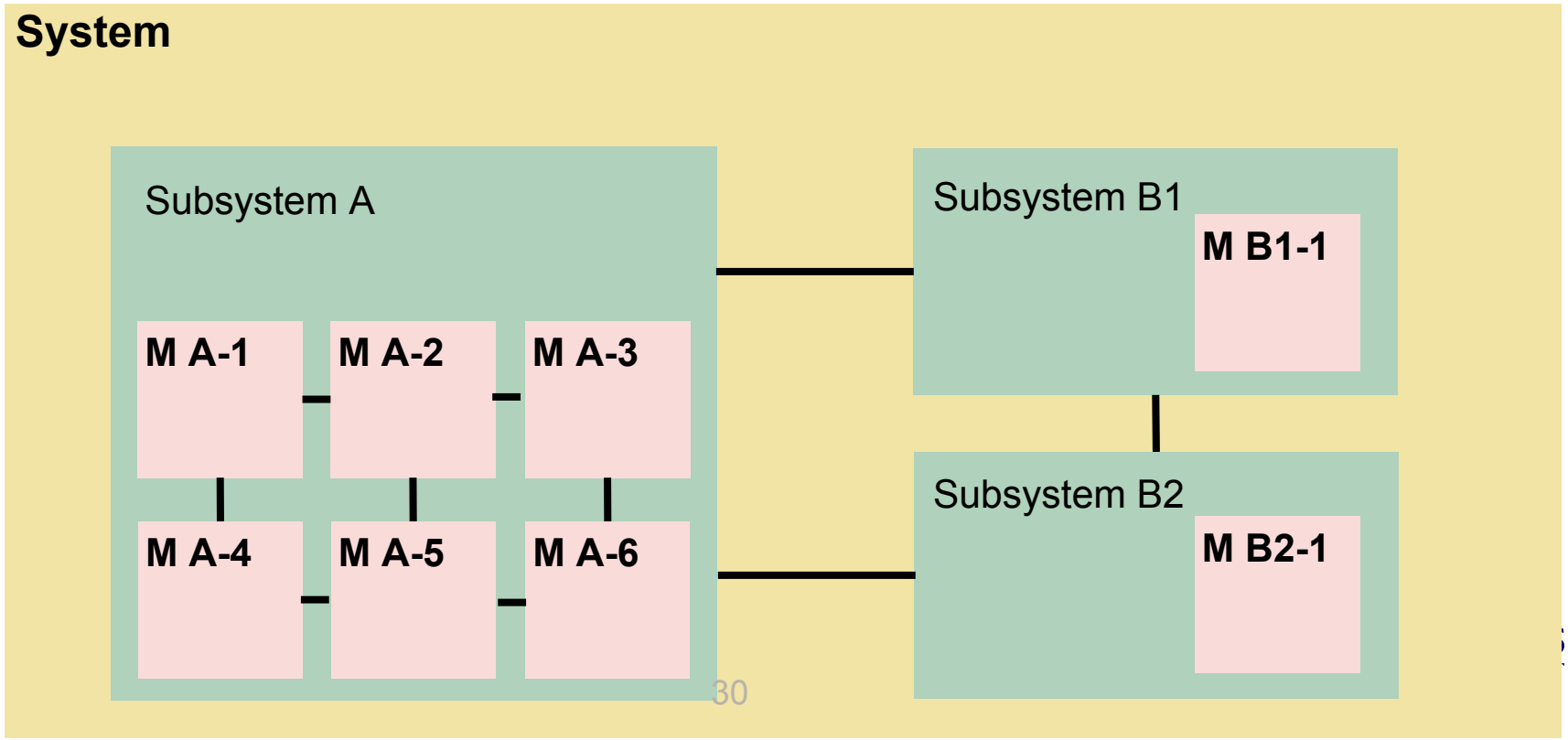
Säkerhet? (safety)

- Att verifiera säkerhetskrav är svårt och dyrt
- Samla alla säkerhetskritiska operationer i separat subsystem



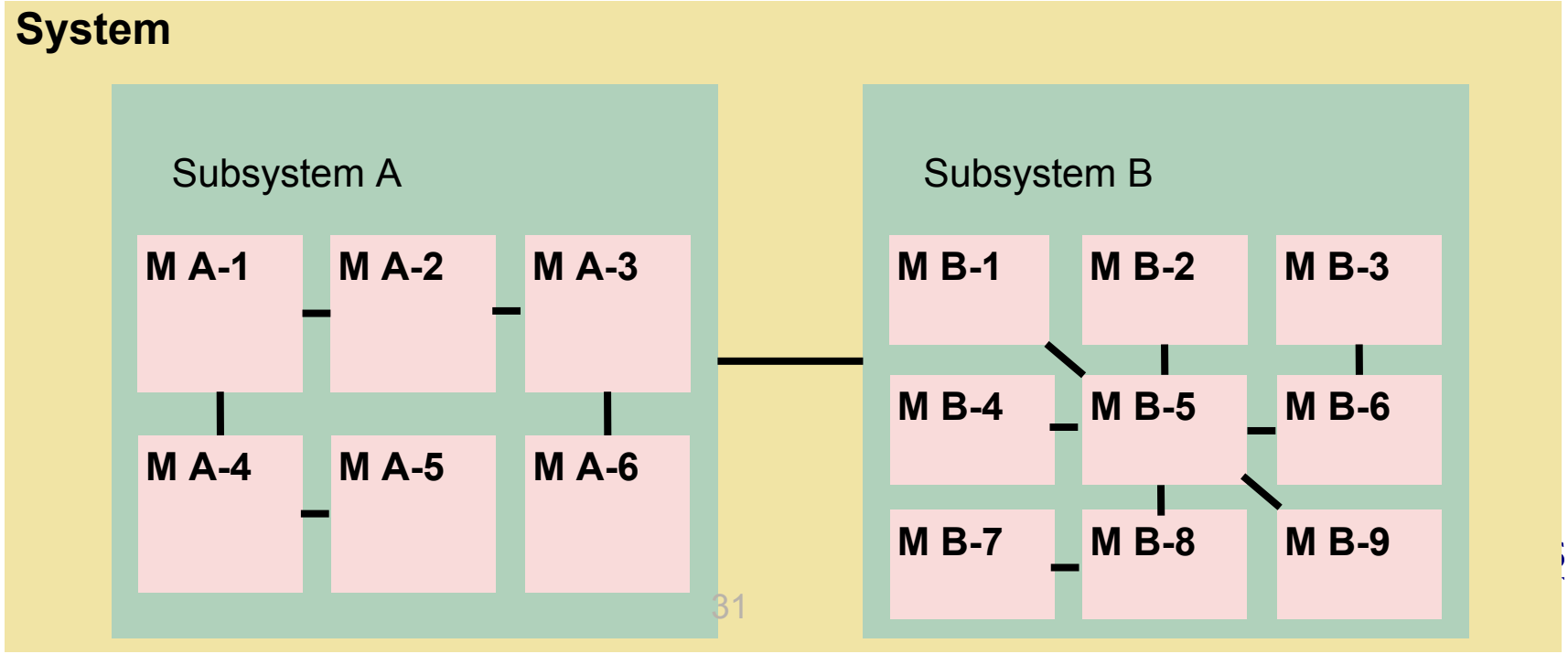
Tillgänglighet?

- Minimera risken att systemet är otillgängligt
- Introducera redundans

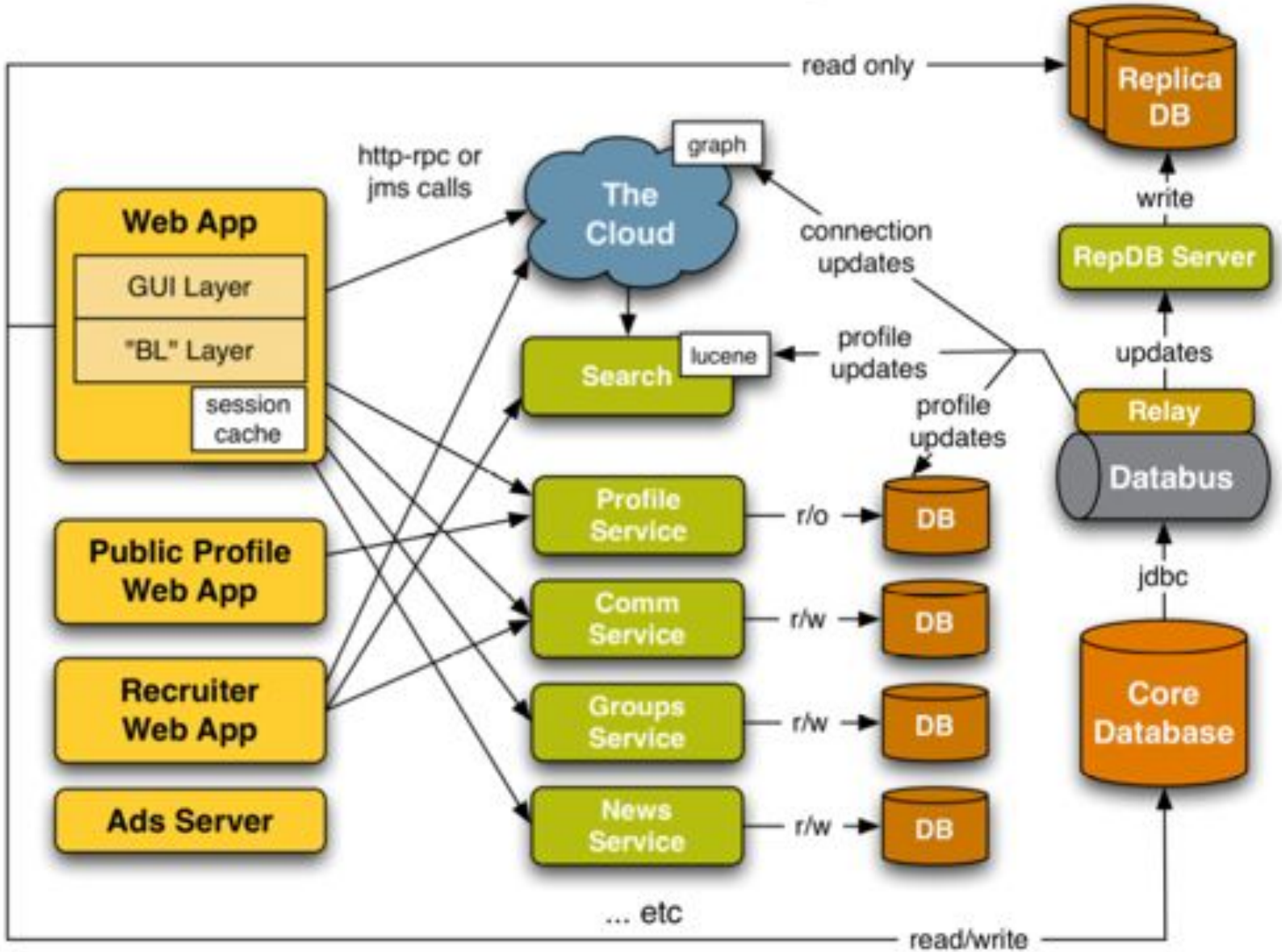


Enkelt underhåll/evolution?

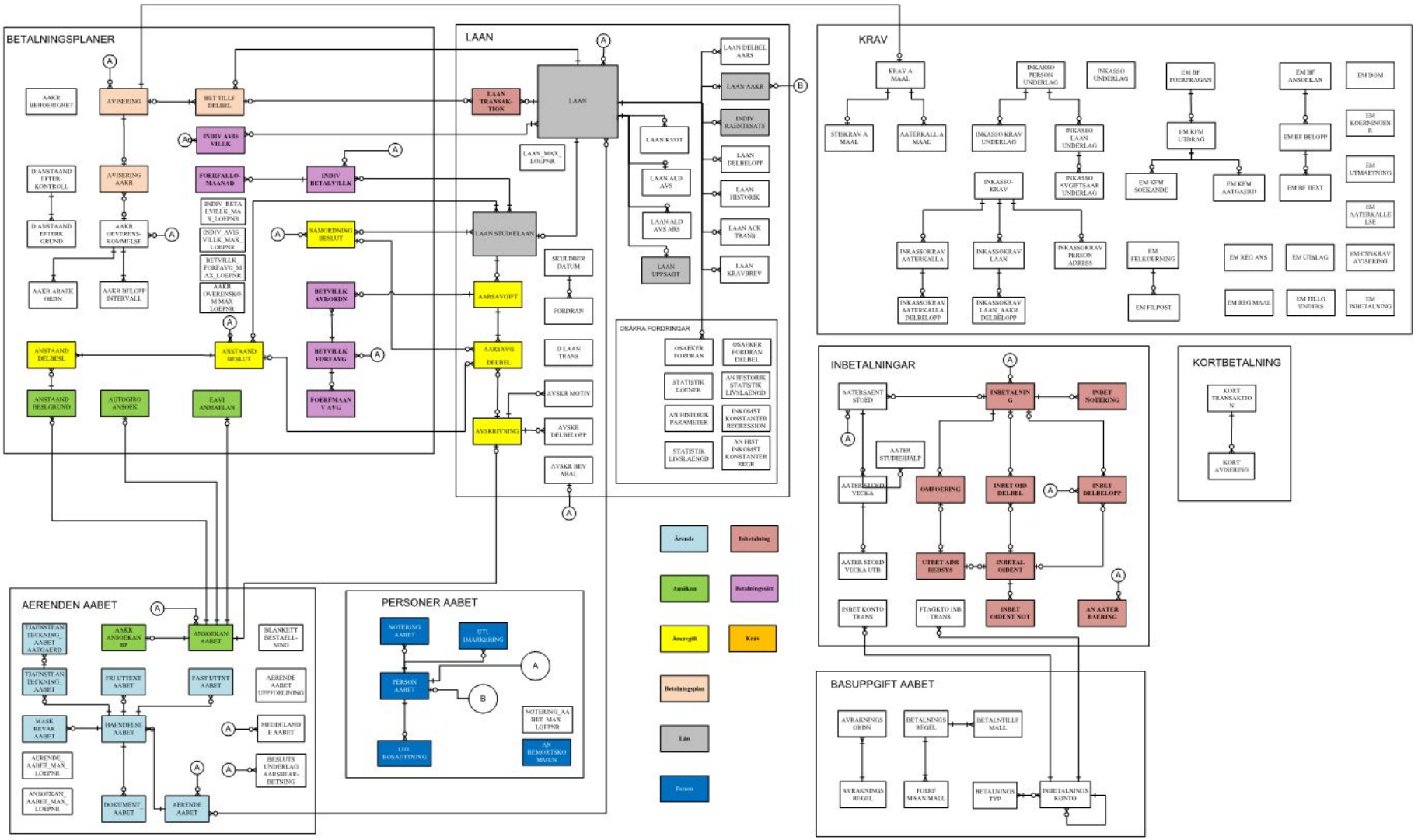
- Fokusera på små oberoende moduler
- Minimal kommunikation gör det lättare att ersätta moduler i framtiden
- Tjänsteorienterad arkitektur (microservices)



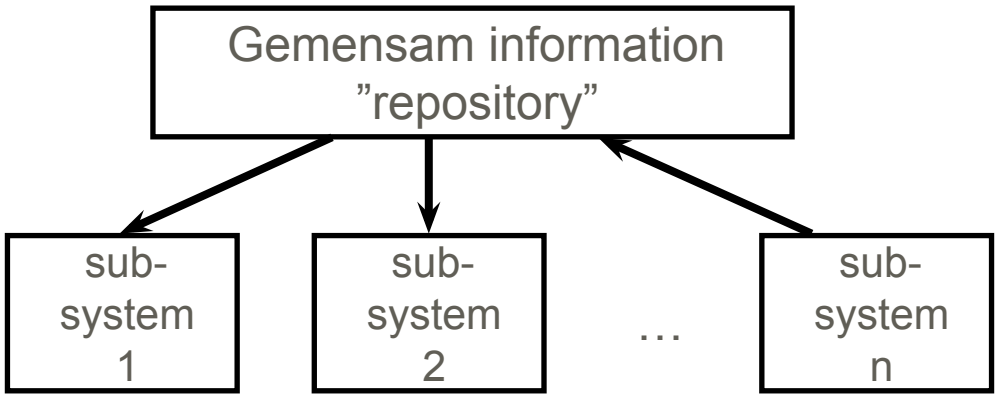
LinkedIn: Java-arkitektur



CSN: Google Drive



Typexempel – Repository (delad data)



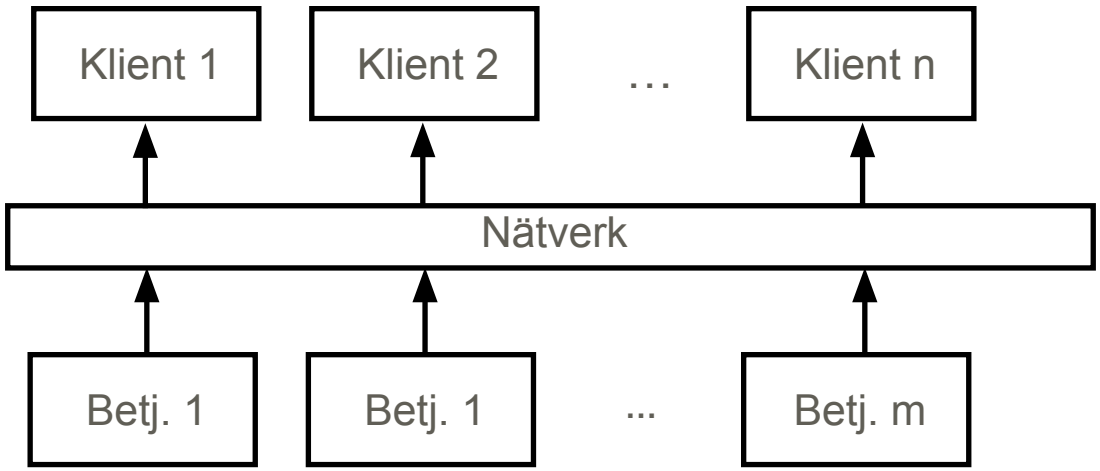
Fördelar:

- Effektivt med mycket data
- Data-producent måste inte veta så mycket om konsument
- Operationer på all data underlättas, t.ex. backup

Svagheter:

- Alla subsystem måste använda samma dataformat
- Vidareutveckling kan vara svårt eftersom mycket bygger på en viss datamodell

Typexempel - Client-server



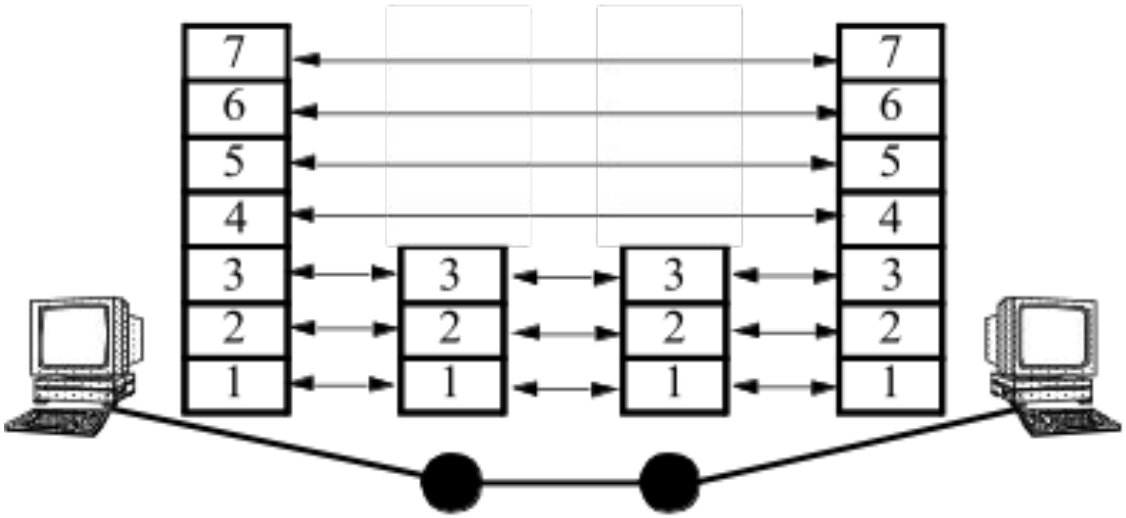
Fördelar:

- Distribuerad arkitektur
- Lätt att lägga till nya klienter och betjänare

Svagheter:

- Uppdatering av klient eller betjänare kan kräva uppdatering av samtliga
- Ingen gemensam datamodell

Typexempel – Abstraktionslager



Varje lager utgör en ”abstrakt maskin” som används av nästa lager

Fördelar:

- Stöd för inkrementell utveckling
- Underlättar portabilitet

Svagheter:

- Kan uppstå beroenden mellan flera lager
- Kan bli sämre prestanda

Design - sammanfattning

Design är både en aktivitet och ett resultat

Arkitekturdesign är en övergripande nedbrytning av systemstruktur: System → Subsystem → Moduler

- Val av arkitektur beror på kvalitetskraven
- Exempel: repository, client-server, abstraktionslager

Objektorienterad design beskriver hur komponenter implementeras av klasser

- Beskrivs vanligtvis med UML
- Sträva efter låg koppling och hög sammanhållning

(Interaktionsdesign utanför kursen)

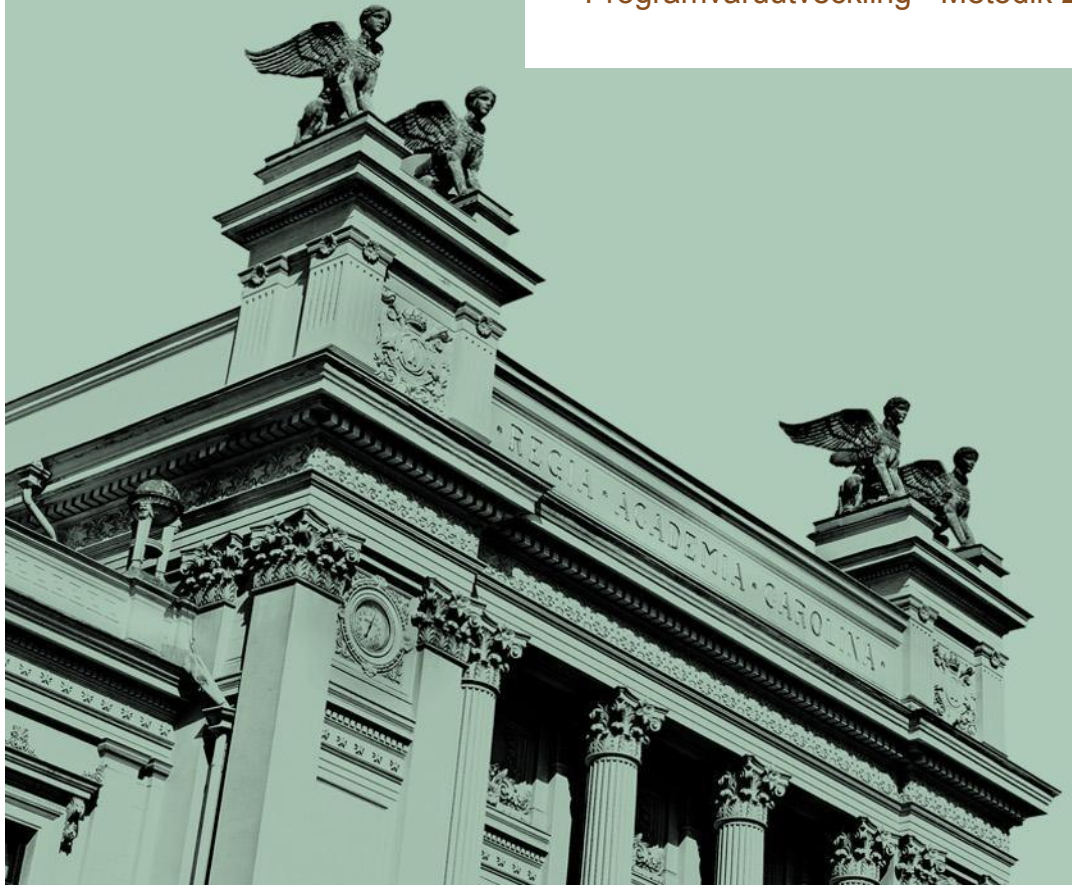
- Grafiska användargränssnitt

LUNDS UNIVERSITET

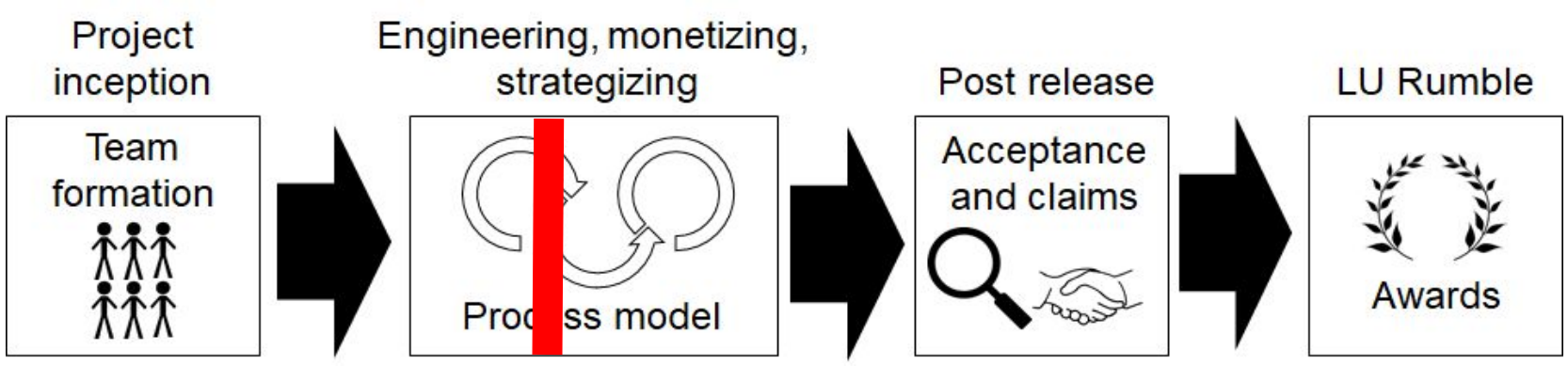
LUNDS UNIVERSITET

Robotmässa

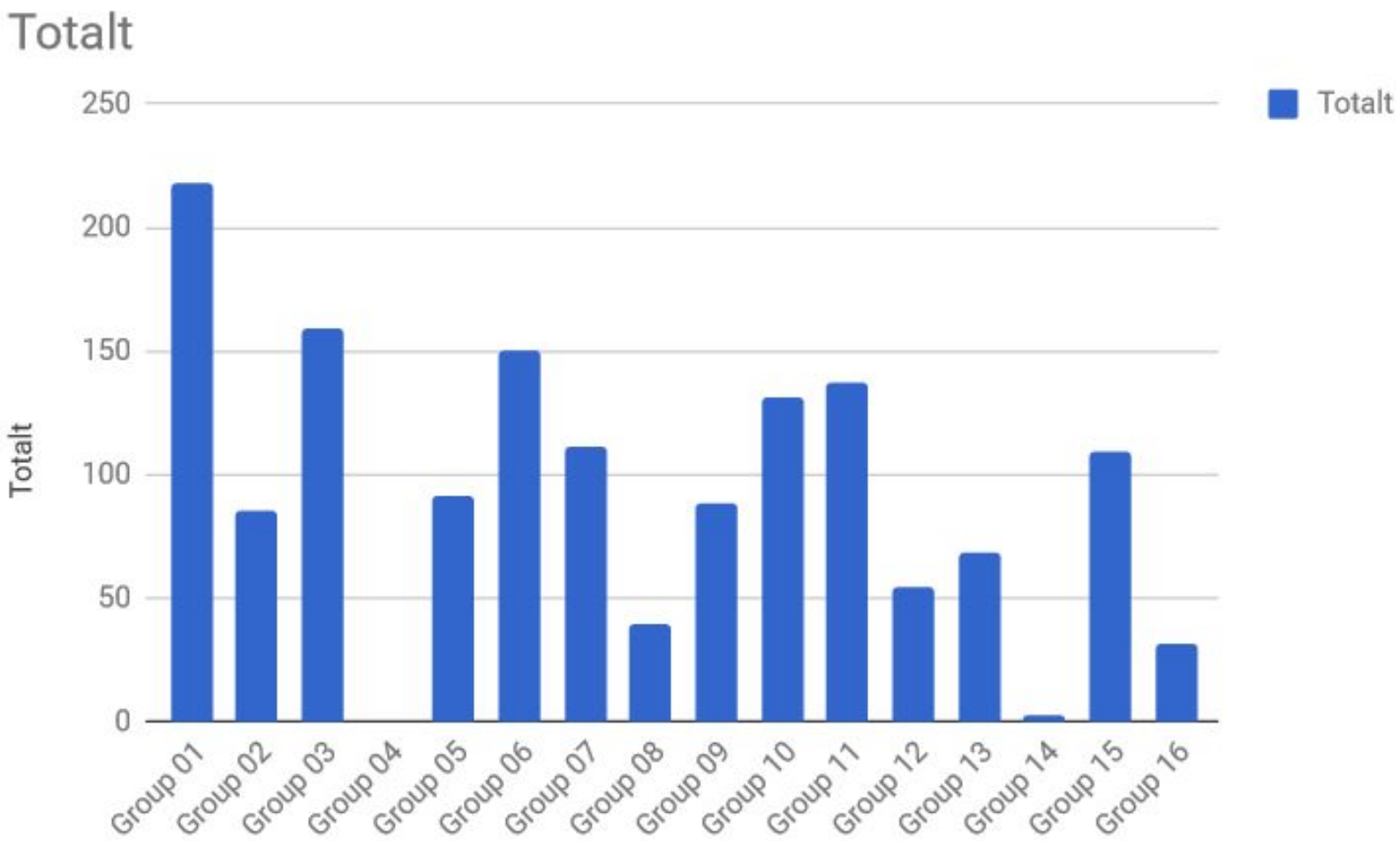
Programvaruutveckling - Metodik 2018 | Markus Borg



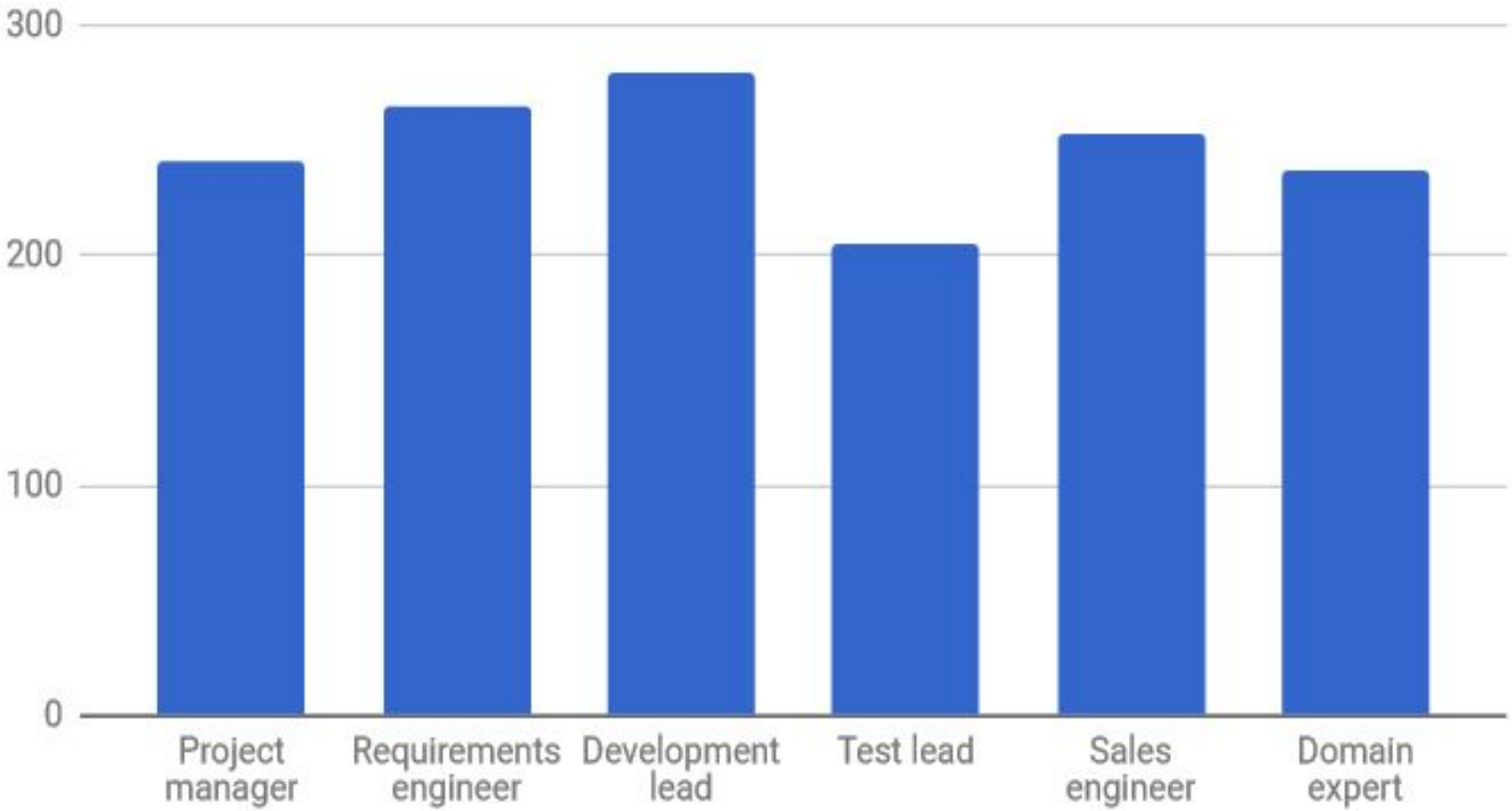
V	tid	måndag 8	måndag 10	måndag 12	måndag 13	måndag 15	måndag K	tisdag 8	tisdag 10	tisdag 12	tisdag 13	tisdag 15	tisdag K	onsdag 8	onsdag 10	onsdag 12	onsdag 13	onsdag 15	onsdag K	torsdag 8	torsdag 10	torsdag 12	torsdag 13	torsdag 15	torsdag K	fredag 8	fredag 10	fredag 12	fredag 13	fredag 15	fredag K	lö	sö
12	Aktivitet	19/3	F1 MA1														Lab1				Ö1											
	Grupp 1-4																Alfa	PW		3308												
	Grupp 5-8																Falk	PW		2116					L1							
	Grupp 9-12																	Alfa		PW	3308											
	Grupp 13-16																	Beta		PW	2116											
13	Aktivitet	26/3	F2 MA1														Lab2				Ö2											
	Grupp 1-4																Alfa			3308												
	Grupp 5-8																Falk			2116						Långfredag						
	Grupp 9-12																	Gamma			3308											
	Grupp 13-16																	Falk			2116											
14	Aktivitet	Annandag påsk						Tentamensperiod						Tentamensperiod						Tentamensperiod						Tentamensperiod							
	Grupp 1-16																															
15	Aktivitet							Tentamensperiod						Tentamensperiod						Tentamensperiod					L2	Tentamensperiod							
	Grupp 1-16																															
16	Aktivitet	16/4	F3 MA1														Lab3				Ö3											
	Grupp 1-4																Alfa			3308												
	Grupp 5-8					L3							L4				Falk			2116												
	Grupp 9-12																	Alfa			3308											
	Grupp 13-16																	Gamma			2116											
17	Aktivitet	23/4	F4 MA1														Lab4				Ö4											
	Grupp 1-4																Alfa			3308												
	Grupp 5-8																Falk			2116										L5		
	Grupp 9-12																	Gamma			3308											
	Grupp 13-16																	Alfa			2116											
18	Aktivitet	Siste april						1 maj									Kod-jour															
	Grupp 1-16																															
19	Aktivitet	7/5	F5 MA1																	Kristi himmelsfärdsdag										L6		
	Grupp 1-16																															
20	Aktivitet	14/5	F6 MA4										L7												L8	Lundakarneval						
	Grupp 1-16																															
21	Aktivitet	21/5	F7 V:A																													
	Grupp 1-16																															
22	Aktivitet													Hemtenta																		
	Grupp 1-16																															
23	Återkoppling																															
24	Återkoppling																															
25	Återkoppling																									Midsommarafton						



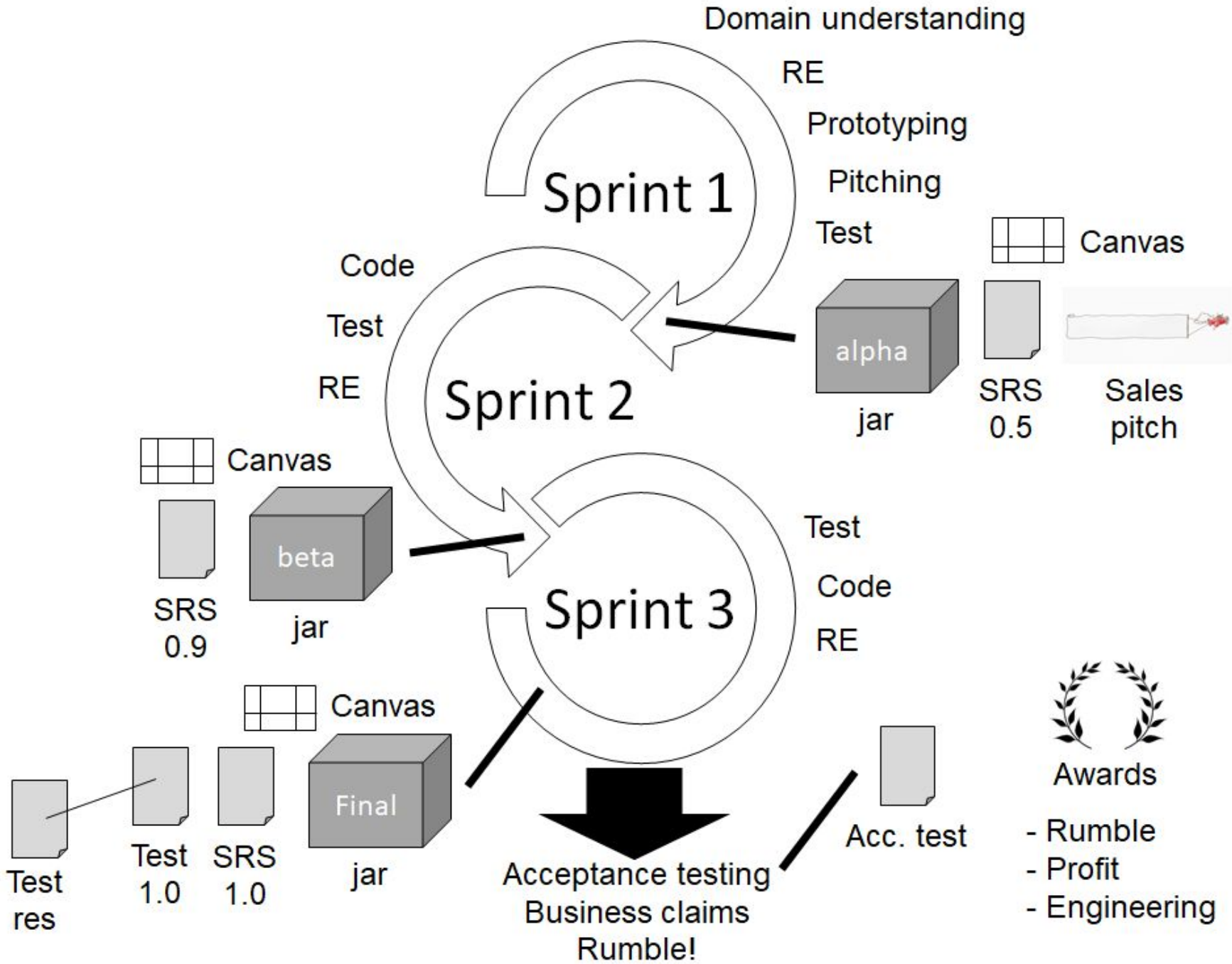
Tidrapporter - Total tid per grupp

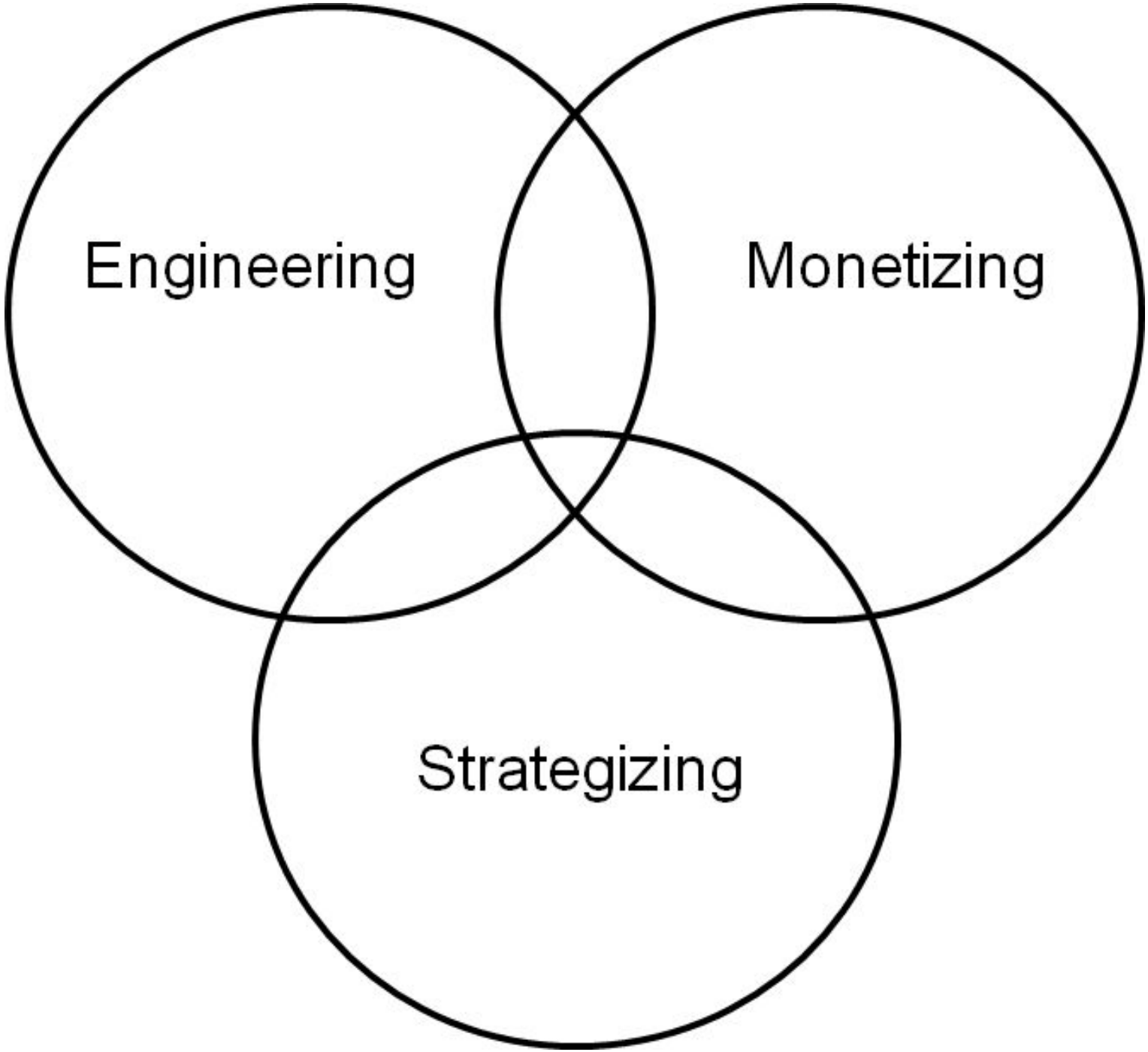


Tidrapporter - Total tid per roll



Flickr: vladislav@munich


Domain understanding
RE
Prototyping
Pitching
Sprint 1
Test
Canvas
Code
Test
RE
Sprint 2
alpha
jar
SRS
0.5
Sales
pitch
Canvas
beta
SRS
0.9
jar
Test
Code
RE
Sprint 3
Canvas
Final
Test
res
Test
1.0
SRS
1.0
jar
Acceptance testing
Business claims
Rumble!
Acc. test
Awards
- Rumble
- Profit
- Engineering


Engineering
Monetizing
Strategizing

Detta har hänt:

- Källkodshantering med git (F2)
- Objektorienterad design (Lab 2)
- Enhetstestning (Lab 2)

Engineering

- Affärsutveckling och features (Ö1)
- Marknadsföring och video pitch (Ö2)

Monetizing

- Grundförståelse för ny domän: Robocode (Lab 1)

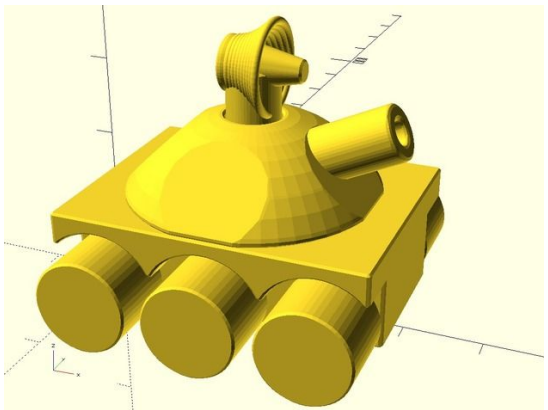
Strategizing

- Automatiserade systemtester (Lab 3)
- Kravspecifikation (Ö3)

- Affärsrelation beställare-leverantör

Engineering

Monetizing



Strategizing

Detta är på gång:

- Budgivning på robotar (L4)

Robotmässa - Filmvisning

Droids

1. Phantom
2. R2D2

Leaders

1. Bob
2. Markov bot
3. Hannibal

Normal bots

1. Rainbow
2. CoboRode
3. DEXTERBOT
4. X-Terminator
5. Wolf Tank
6. Secubot
7. SHARP
8. Rut the Ram
9. Mr. robot
10. T.R.A.C.I.E
11. DopeBot

LUND UNIVERSITY

L4: Imorgon kl 23.59

Inlämning av inköpsvektor

- Alla grupper ska lägga bud på samtliga (övriga) gruppers robotar
- Alla grupper har €100
- Minsta bud är €10 (Motsvarande ETSA02 Basic Droid)
- Inköpsvektorn ska skrivas under av domänexperten och lämnas i Grå Skåpet
- Exempel:

Inköpsvektor Grupp 02	
Grupp 01 - <ROBOTNAMN>	€15
Grupp 02 - <ROBOTNAMN>	N/A
Grupp 03 - <ROBOTNAMN>	€10
...	
Grupp 16 - <ROBOTNAMN>	€22

LUND UNIVERSITY