Föreläsning 3:
Test, Konfigurationer

Ingenjörsprocessen metodik ETSA01 VT15 | Jonas Wisbrant

Detta har hänt....

Pratat och skapat krav och plan

Övning 2 Riskhantering, intressenter och kravgranskning.

Projektet har granskat Krav 0.9 och reviderat utifrån identifierade problem inför krav 0.99?

Jonas Wisbrant - kort CV

Samhällsvetare vid LU	1989
Kommunikation och webbutveckling	1990
Programvaruingenjör LTH	2002
Institutionen för Datavetenskap 1	2002
LUCAS - Center for Applied Software Research	
Diverse undervisning	
Det Norske Veritas	2008
Institutionen för Datavetenskap 2	2009
EASE / Programvaruportalen / kommunikation	
Datorer i System	
Ingenjörsprocessen	
Datavetenskap + LU-webb	2011
Datavetenskap + LTH-webb	2013
Datavetenskap, LTH-webb EGNA Studier	2014-2016

Kursombud - nytt försök

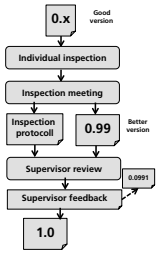
Beslut:

D - Rasmus Göransson,
D - Oscar Sigurdsson,
C - Måns Ansgariusson,
I - ???

Kursinformation

V 16:

- **Nu:** Föreläsning - Testning, Konfigurationer
- **Idag kl 24:** Projektplan + krav 0.99 + föregående granskning
  - ICE: Projekthandledare
- **Övning 3:** Test
  - Om testfall: T.1-8 (i förväg)
  - Påbörja testplan: T.15-16
  - Återkoppling på 0.99 och projektplan fb3



V 17:

- **Må kl 10:** Föreläsning - Arkitektur mer om test, demo
- **Ö4:** Mer om test

Hög tid att kolla på koden

Downloads på:
http://cs.lth.se/kurs/etsa01/projekt2014/haardvarugraenssnitt_och_drivrutiner/

Agenda

Verifiering & Validering

- Vad är testning?
- Testprocessen
- Testtekniker

Konfigurationshantering

- står inte så tydligt i boken
- regler för projektet finns i projektmaterialet på kurswebben

Produktlinjer

- står inte så tydligt i boken (kort i mån av tid)

Varför trillade den första Ariane V-raketen ned?

Vad kan gå fel?

Vad brukar gå fel?

Varför då?

Vad kan man göra åt det?

Ariane 5 Flight 501 Failure—A Case Study of Errors
Ken Robinson School of Computer Science and Engineering University of New South Wales, 16th December 1996 Revised: 22rd March 2011
http://www.di.unito.it/~damiani/ariane5rep.html

Verifiering & validering

Ingenjörsprocessen metodik ETSA01 VT15 | Jonas Wisbrant

Verifiering & Validering

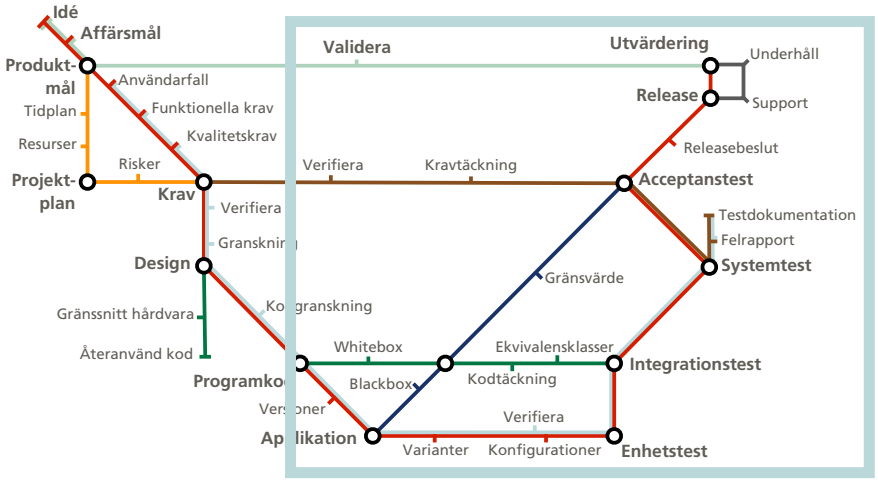
Verifiering

Bygger vi produkten rätt?
Följer vi kravspecifikationen?

Validering

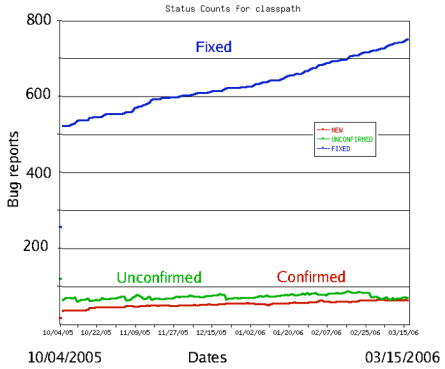
Bygger vi rätt produkt?
Kommer beställaren att bli nöjd?
När beställaren sina affärsmål?

Verifiering & Validering

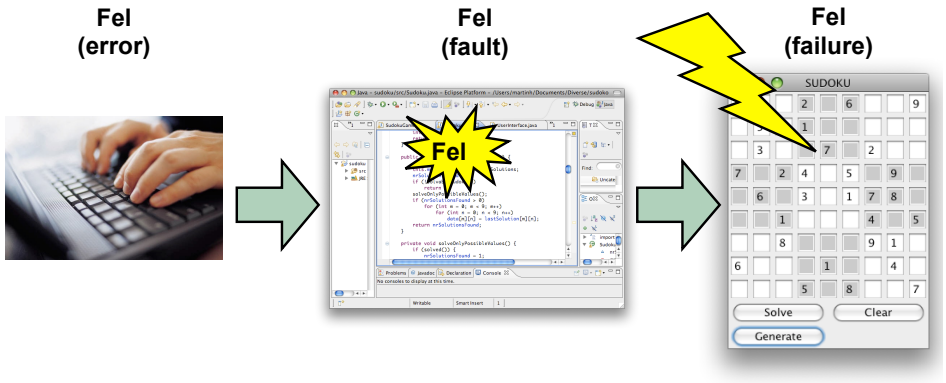


Varför testar man?

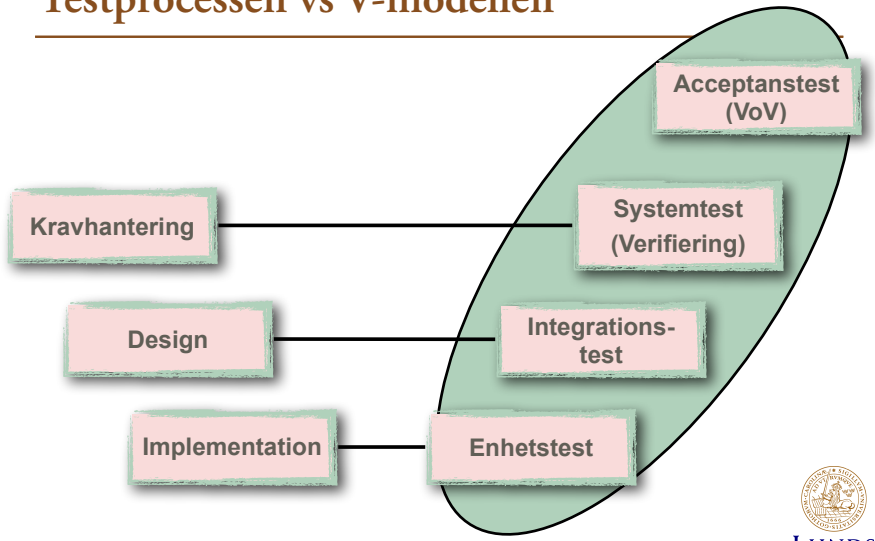
- Man vill hitta fel för att man vill förbättra produkten
- Man vill visa att produkten är bra
- Man vill ta reda på hur många fel som finns i produkten för att man vill veta hur bra produkten är.
- …men oavsett vilket så kan man aldrig visa att det inte finns fel, bara att det finns fel…



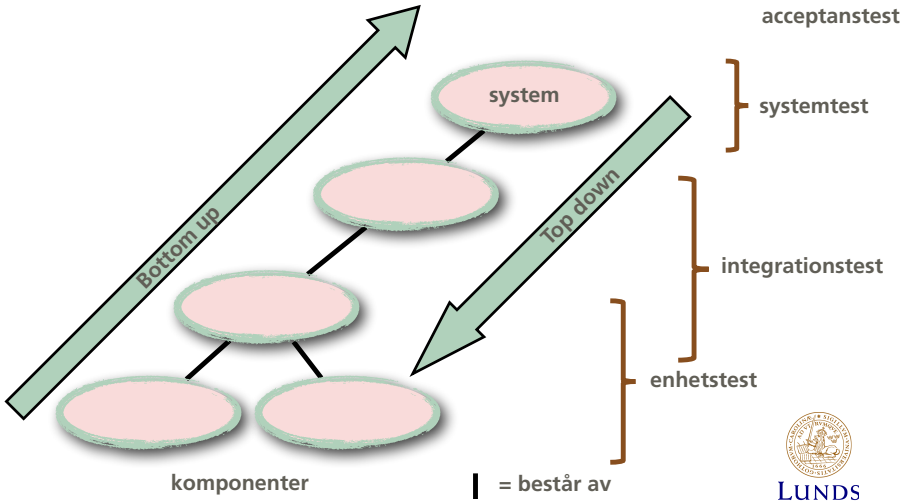
Fel, fel, och fel…



Testprocessen vs V-modellen

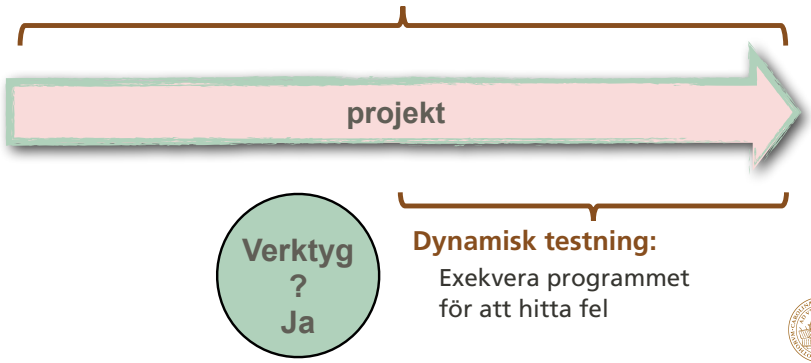


Testprocessen vs. Design och Integration



Dynamisk vs statisk testning

Statisk testning (Granskning):
Hitta fel utan att exekvera programmet



Dynamisk testning:
Exekvera programmet
för att hitta fel

Enhetstest
- testfall för procedurer/metoder

För att testa procedur F(x1,...xn), gör följande:

- Sätt tillstånd till rätt värde
- Definiera värden för parametrar x1,…xn
- Anropa Svar = F(x1,...xn)
- Jämför Svar med rätt resultat
- Registrera om resultatet var korrekt eller inte

Kan stödjas med verktyg, t ex CuTest, CUnit,…

Med objektorientering kan t ex JUnit användas

Integrationstest / Systemtest

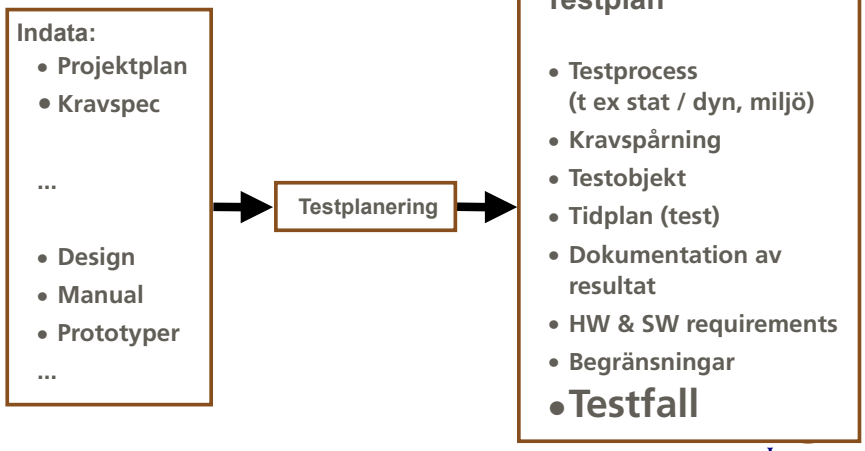
Testfall bestäms t ex baserat på specifikationer av gränssnitt i design

Test utförs i samband med integration av olika delar av systemet

Testfall måste utföras för varje del som integreras
--> automatisering önskvärd

Regressionstest = upprepad testning vid ändring av programvara
Kan utföras under utveckling och under underhåll

Testplanering



Testfall, två exempel

Test case: Buy cola

Pre-condition:
Machine in start state.

Post-condition: Machine in start state. A cola can received.

1. Press the cola selection button.
2. Insert a 10kr coin.
3. Insert a 5kr coin.
4. Receive a cola can.

Test case: Press more than one button

Pre-condition: Machine in start state.

Post-condition: Machine in start state. A water can is received.

1. Press the water selection button.
2. Press the beer selection button.
3. Insert a 5kr coin.
4. Receive a water can.

Spårbarhet mot krav

Alla krav ska testas av minst ett testfall

Alla testfall ska testa minst ett krav

	Test case											
Req	1	2	3	4	5	6	7	8	9	10	11	12	13
1													
2													
3													
4													
5													
6													
7													
8													
9													
10													
11													
12													
13													

Detta kan t ex visas i en matris (krav x testfall) i testplanen

Black-box vs. White-box

Black-box

- Programet 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

White-box

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

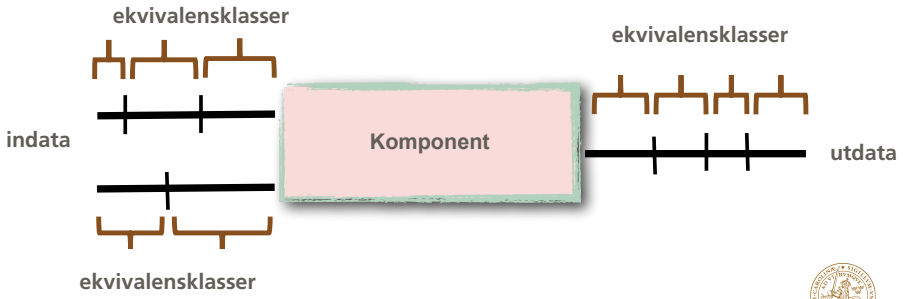
”Black-box”-test

Programkoden 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

Ekvivalenspartitionering

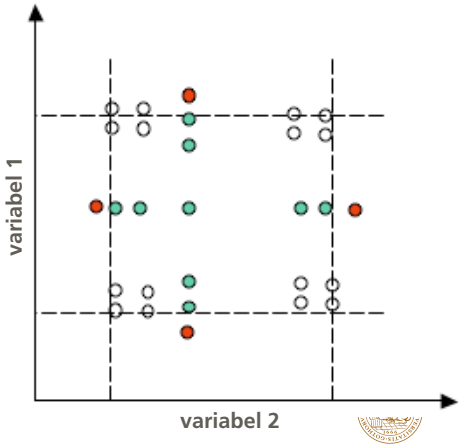
Hitta värden för in och utdata som behandlas på inbördes enhetligt sätt



Gränsvärdestestning

För n variabler, ett intervall var

- **Vanlig gränsvärden:** innanför eller på gränser => 4n+1 testfall (gröna)
- **Robust-test:** även utanför gränserna => 6n+1 testfall (röda)
- **Worst-case:** även alla kombinationer av gränsfall (men inte robust-test): 5^n testfall (vita)



Parvis testning

Vissa fel "single mode"

- T ex lägga in viss typ av kurs

Andra fel uppstår som kombination av två parametrar:

- T ex lägga in viss typ av kurs för viss institution
- Parvis testning: täck alla möjliga kombinationer av värden för alla möjliga par av parametrar

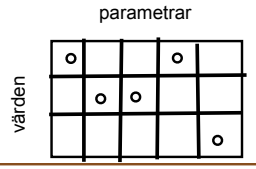
Eller ännu fler parametrar

- T ex lägga in viss typ av kurs för viss institution för visst år och viss studentgrupp

Antag ett system för att hantera kurser och studenter, med följande parametrar:

- Kurstyp (G1, G2, A)
- Institution (CS, EIT, Math,…)
- År (2012, 2013, 2014, 2015)
- Studentgrupp (C, D, E,…)

Parvis testning



n parametrar med m möjliga värden vardera ger m² par av värden för varje par av parametrar

Första med resten ger m²(n-1) par, andra med resten m²(n-2) par, osv

Antal par = $$\sum_{k=1}^{n} m^2(n-k) = m^2 \sum_{k=1}^{n} (n-k) = m^2 n \frac{(n-n)+(n-1)}{2} = m^2 n(n-1)/2$$

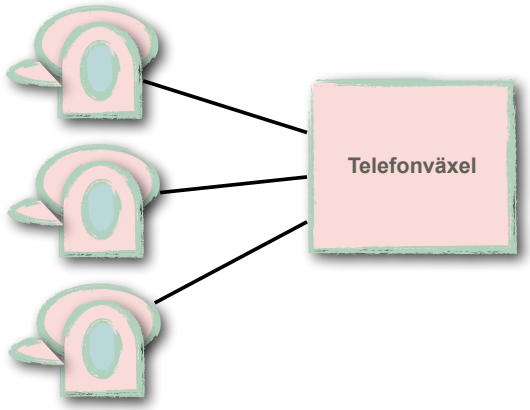
Varje testfall täcker $$(n-1)+(n-2)+\ldots+1 = \sum_{i=1}^{n-1} i = \sum_{i=0}^{n-1} i = n(n-1)/2$$ **par**

Om varje testfall täcker olika par krävs det alltså m² testfall

- I praktiken omöjligt ha unika par för varje testfall
- och olika antal värden för olika parametrar

Stresstestning

Kontrollera vad som händer vid hög belastning, t ex:



 # telefoner > max
 # samtidiga samtal > max
…

ERROR: ArrayOutOfBounds ?

Genomförande av test och rapportering

I simulerad miljö
- endast i utvecklingsmiljön

I mer verklig miljö
- i testuppsättning

I verklig miljö

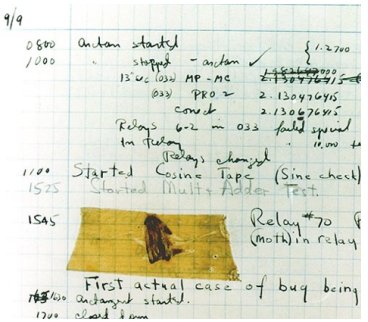
Alla fel rapporteras/registreras
- Testfall
- Resultat
- Feltyp
- Allvarlighet
- Ev ytterligare beskrivning

Tänk kommunikation:
- mellan individer och organisationer
- över tiden

Tänk dokumentation:
- Vad fungerar?
- Vad fungerar ännu INTE?

Testprotokoll

Redogör för resultatet av test
- t.ex. en tabell med testfall, testresultat datum, etc



TESTPROTOKOLL

Testfallsnr.	Korrekt ja/nej	Position i testfall	Feltyp	Grad	PR nr.

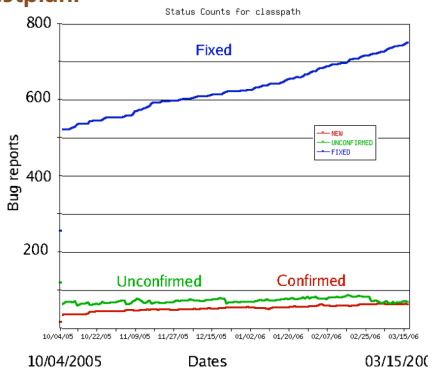
Skillnader och likheter mellan en testrapport och ett granskningsprotokoll?

Färdigtestat?

När vi gjort vad vi lovat i projekt- och testplan:

t ex visat att alla kraven på alla abstraktionsnivåer är uppfyllda:
- ...traverserat alla grenar i koden...
- ...med alla upptänkliga kombinationer av variabler...
- ...på alla plattformar...
- ...med max+ belastning

Tillräckligt bra?



Sammanfattning - Test

Testning kan påvisa fel, men inte bevisa att det inte finns fel

Testprocessen kopplar samman integration med testning: enhetstest, integrationstest, systemtest, acceptanstest

Testmetoder kan vara antingen ”black box” eller ”white box”

Dokumentgranskning är en testform där man kritiskt läser ett dokument på ett systematiskt sätt.

Verifiering	Acceptanstest	Gränsvärdestestning
Validering	Testfall	Parvis testning
Dynamisk testning	Kravtäckningsmatris	Stresstest
Statisk testning	Black-box-test	Code coverage
Enhetstest	White-box-test	Bransch coverage
Integrationstest	Ekvivalensklass-partitionering	Testprotokoll
Systemtest		Felrapport

LUNDS UNIVERSITET

Konfigurationshantering

Ingenjörsprocessen metodik ETSA01 VT15 | Jonas Wisbrant

Diskussion:
Försök förklara för din granne:

Vad är det för skillnad mellan:

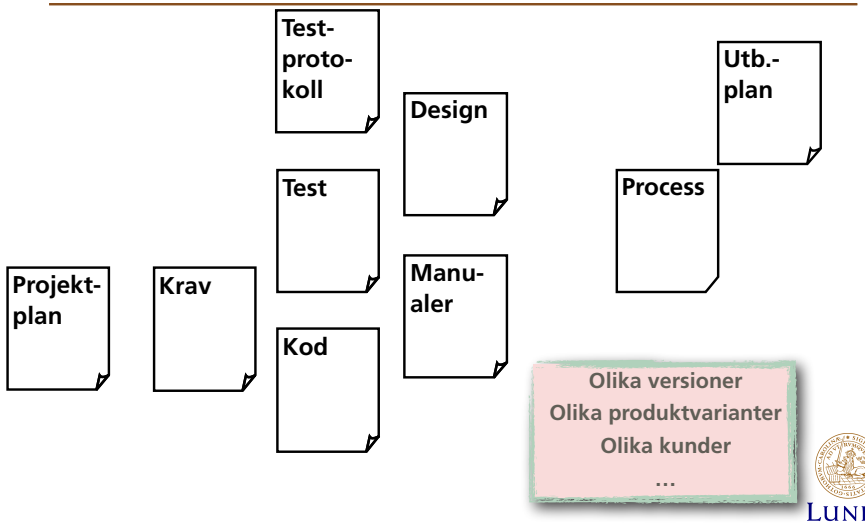
Konfiguration

Version

Variant?

Hur håller vi isär pågående versioner från ”frysta”?

Konfigurationshantering



Configuration Management

Configuration management (CM) is a field of management that focuses on **establishing** and **maintaining** consistency of a **system's** or **product's** performance and its **functional and physical attributes** with its **requirements, design, and operational information** throughout its **life**.

Configuration Management

Configuration management (CM) is a field of management that focuses on **establishing** and **maintaining** consistency of a **system's** or **product's performance** and its **functional and physical attributes** with its **requirements, design, and operational information** throughout its **life**.

Typiska frågor man kan vilja ha svar på

Vilka kunder har installerat en viss version av systemet?

Vilken hårdvara och OS krävs för en viss version?

Vilka versioner har levererats av ett visst system?

Vilka versioner påverkas om jag ändrar en viss komponent i systemet?

Hur många olika ändringar håller man på att göra av ett visst system?

Hur många rapporterade men ännu ej rättade fel finns det i ett system hos en viss kund?

Planering av konfigurationhantering

Bestäm vad som ska konfigurationshanteras

- Projektdokument – inte säkert
- Produktdokument – ja
- Organisationsdokument – ja, men inte av projektet

Bestäm rutiner för konfigurationshantering

Bestäm ansvarig för konfigurationshantering

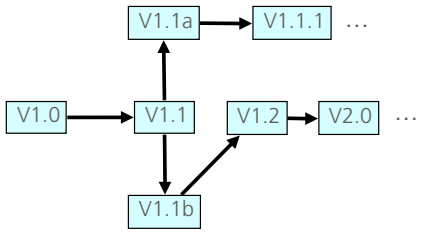
Bestäm hur ändringshantering ska gå till

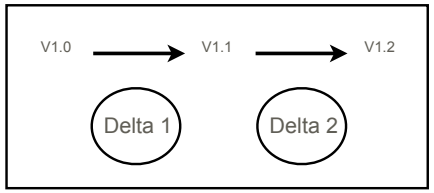
Bestäm vilka verktyg för konfigurationshantering som ska användas (och vilken typ av databas man ska använda)

Versioner, releaser och delta

V 1.0 release
V 1.1 version
V 1.2 version
V 1.3 version
V 2.0 release
V 2.1 version
… …

ID för varianter/grenar





Lagra ändringar i delta

Hålla reda på ändringshistoria

Ändrat till:	
1.1	Tidigarelagd deadline för I2 och L3 för projektgrupper 1-10
1.1	Mindre korrekturändringar
1.2	Bytt plats på ÖL för grupper 1-2 och 28-29
1.3	Grupp 31 tillagd
1.3	Ny lokal för grupperna 11-15 och grupperna 26-31 torsdag morgon
1.4	Tydliggjort schema angående Ö1a, PW och Ö1b
1.4	Tydliggjort schemat för Ö4a, PW och Ö4b

Ver.	Date	Resp.	Description
1.0	070921	BO	Baseline
1.1	071010	EA	Removed requirement about coin types
1.2	090209	EA	Updated use cases
1.3	130131	KW	Updated introduction and some minor other changes
1.3	130305	JW	Enumerated high level goals and constratints

Delta ?

Ändringshantering ≈ ordnad övergång

```
Begär ändring genom att fylla i formulär
Analysera ändringsbegäran
Om ändring nödvändig och korrekt {
        Utvärdera hur ändring kan göras + kostnad
        Skicka förfrågan till CCB
        Om ändring accepterad {
                Ändra i systemet
                Uppdatera ändringsbegäran
                tillverka ny version av systemet
        }
}
```


Formulär för ändringshantering, del av information

Del 1
- Namn
- Projekt
- Ändring nr
- Anledning
- Objekt att ändra

Del 2
- Utredare
- Resultat av utredning

Del 3
- Förslag på ändring
- Skattad tid
- CCB-möte, datum
- CCB beslut

Del 4
- Ansvarig för ändring
- Kommentar
- Ny version:

CCB – Change Control Board

Strategiska beslut om vilka ändringar som ska ske

Inte bara tekniska beslut

Representerar kund, projekt och andra intressenter

För programvara som används i flera produkter blir det mer komplicerat

Systembygge

Bygga en fungerande version av hela systemet från de komponenter som finns, t ex

- Senaste versionerna av allt
- En viss version av en viss produkt
- En viss version av en viss produkt till en viss kund
- En viss version av en viss produkt till en viss plattform
- …

Behövs t ex vid systemtest och andra tester (och såklart vid leverans)

Tekniska och politiska versioner?

Teknisk version varje gång man sparar

- praktiskt men inget man fattar beslut kring

Politisk version när man ändrar nummer:

- Låser historiken
- Underlag för beslut
- Beskriver förändringar
- Kopplar ihop olika dokument
- Möjliggör ordnad förgrening

Versionshistoria Kravspecifikation

Version	Datum	Ändrat av	Beskrivning
0.0	2010-03-21	AA	Start
0.1	2010-03-23	AA	Funktionella krav
0.2	2010-03-28	BB	Kvalitetskrav krav
0.3	2010-04-12	AA	Mer om GUI
0.99	2010-04-19	AA	Extern granskn.
1.0	2010-04-25	AA	**MS1**
1.1	2010-04-27	BB	CR Req1.0.1
1.2			

Ändringshantering i ert projekt

Baserat på:

- Efter baseline – dvs efter att milstolpe uppnåtts
- Ansvariga för dokument som bestämmer vem som får ändra vad
- Uppdatering av versionsnummer
- Spridande av information inom gruppen
- Speciella regler för kravspecifikationen (efter baseline får man inte ändra utan att ansöka hos projekthandledaren)

Kort version av vår process:

1. Person A finds out that there is something wrong in a document
2. Person A locates the fault in the document (currently in version 1.0)
3. Person A notifies the person responsible for the document (person B) and asks for permission to change it
4. Person B gives person A permission to change the document
5. Person A updates the document, updates the version number to 1.1 and gives it to person B.
6. Person B is now responsible for telling all persons that are working with the document or depend on it, that there is a new version.

Versionshistoria i dokument och ändringsformulä

Versionshistoria Kravspecifi

Version	Datum	Ä
0.0	2010-03-21	A
0.1	2010-03-23	A
0.2	2010-03-28	B
0.3	2010-04-12	A
0.99	2010-04-19	A
1.0	2010-04-25	A
1.1	2010-04-27	B
1.2		

change_request_form.txt - Notepad

File Edit Format View Help

```
*** Part A - filled out by change requester:

1. Project group number:
2. Change request number:
3. Change requester (Name):
4. Document and document version:
5. Date:
6. Requested change (max 500 words)

*** Part B - filled out by projec

7. Decision:
   ( ) i change accepted
   ( ) ii change not accepted
   ( ) iii change partly accepted:
8. Date:
9. Name:
10. Comment (optional):
```

Eller implementera detta i driven i samråd med handledare

Dvs, era versionsnummer för kravspecifikation blir t ex

0.0	första version – inom projektet
0.x	nya versioner efter arbete med dokumentet
0.99	version inlämnad till projekthandledare
1.0	dokument för godkänd milstolpe
1.1	ny version efter att projekthandledare godkänt ändring
1.2	--"--

- Innan 0.99 gör ni som ni själva vill
- Från och med 1.0 ska ni följa procedurer för ändringshantering

Konfigurationshantering - sammanfattning

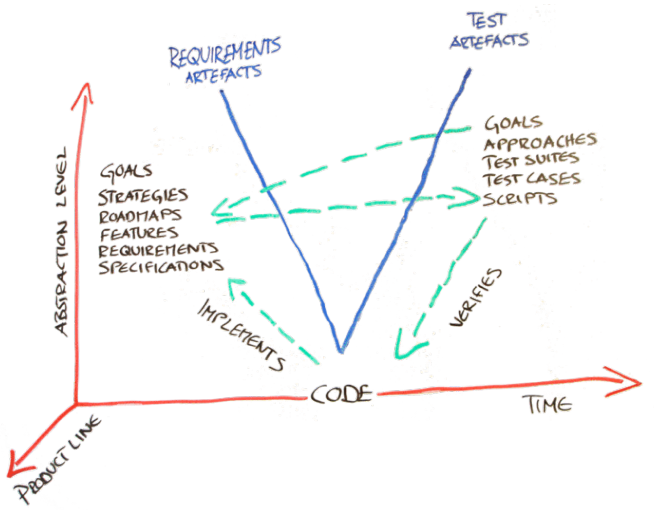
Ändringshantering görs för att rätt beslut ska tas baserat på både tekniska frågor och affärsbeslut

I projektet får ni göra mycket som ni vill innan v 0.99, men efter det måste ni följa procedurerna i kompendiet. För kravspecifikationen måste ändringar efter 1.0 godkännas av projekthandledaren.

- Innan 0.99 gör ni som ni själva vill
- Från och med 1.0 ska ni följa procedurer för ändringshantering

Plattformar & produktlinjer

Ingenjörsprocessen metodik ETSA01 VT15 | Jonas Wisbrant



Plattformar inom produktutveckling

Inte bara inom programvara:

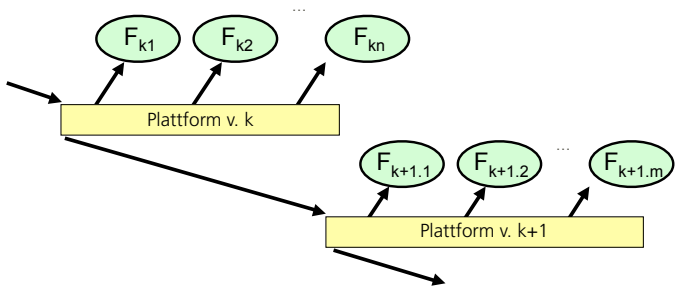
Black & Decker

- Battery Pack
- Motor Pack …

Bilindustri
Peugeot, Fiat, Toyota, etc …"

Software product line

A software product line is a set of software-intensive systems that share a common, managed set of feature that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.



Vad innebär detta?

Parallell utveckling -> lägre kostnader, fler produkter, kortare tid till marknad per produkt

Ökad komplexitet: t ex nya krav och påverkar flera produkter (och plattformar). Mer komplexa projektplaner, testning, organisation

Långsiktiga beslut för plattform, kortsiktiga beslut för produkter

Nya produkter innehåller funktioner från tidigare produkter -> I stort sett aldrig specifikation "från 0". Mycket "arv"

…

Kursinformation

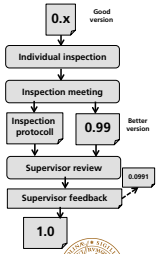
Hög tid att kolla på koden för garaget

V 16:

- **Nu:** Föreläsning - Testning, Konfigurationer
- **Idag kl 24:** Projektplan + krav 0.99 + föregående granskning
  - ICE: Projekthandledare
- **Övning 3:** Test
  - Om testfall: T.1-8 (i förväg)
  - Påbörja testplan: T.15-16
  - Återkoppling på 0.99 och projektplan fb3

V 17:

- **Må kl 10:** Föreläsning - Arkitektur mer om test, demo
- **Ö4:** Mer om test



LUNDS UNIVERSITET