

## Kan vi stole på datamaskinen?

Sikkerhet og systemfeil  
Kapittel 4 - A Gift of Fire

Kirsten Ribu 14.03.2007

HIO -Anvendt datateknologi - Kirsten Ribu 2007

1

## Innlevering i slutten av måneden - prosjektet

- **Leveranse 1 – på papir!**
- **Leveres i Kirstens skap i 2. etasje.**
- Utfyllt **prosjektplan:**
  - Ha etablert kontakt med bedrift, ha fordelt arbeidsoppgaver og indentifisert risiko.
  - Beskriv bedriften, og datasystemet dere skal studere.
  - Prosjektplanens del 1. 2. og 3. skal leveres. Husk versjonsnummer og navn på gruppe og prosjektet.  
**Fredag 30. mars kl 1600**

HIO -Anvendt datateknologi - Kirsten Ribu 2007

2

## Ukeoppgavene

- Alle ukeoppgaver leveres på Fronter i **ett dokument**
- **Mappen Innlevering benyttes.**
- **Fredag 23.mars kl 1600**

HIO -Anvendt datateknologi - Kirsten Ribu 2007

3

## I dag: Systemfeil og testing

- Validering og verifisering
  - **Å sørge for at et datasystem tilfredsstillers brukernes behov**
  - **Kvalitetskontroll**
  - **Avdekking av feil**

HIO -Anvendt datateknologi - Kirsten Ribu 2007

4

## Hva er sikkerhetskritiske systemer?

- Systemer som kan true menneskers **liv og helse** (kuvøser, medisinsk utstyr)
- Systemer som kan føre til **enorme økonomiske tap (konkurs)**
- Systemer som kan **true miljøet** (eksplosjoner)
- **Bransjer:** Medisin, romfart, jernbane, fly, nukleær, ...

HIO -Anvendt datateknologi - Kirsten Ribu 2007

5

## Samfunnskritiske funksjoner

- Energi og kraftforsyning
- Transport
- Elektronisk kommunikasjon
- Mat og vann
- Avløp og renovasjon
- Sosial- og helsetjenester
- Nødtjenester
- Forsvar
- Storting og regjering
- Sentral, regional og lokal administrasjonen
- Industri
- Arbeidskraft
- Lov og orden
- Bank- og pengevesen

HIO -Anvendt datateknologi - Kirsten Ribu 2007

6

## Robust kraftforsyning?

- Høsten 2003:
  - Italia mørklegges
  - London mister strømmen
  - Nordlige deler av USA/
  - Sørilige deler av Canada mørklagt
  - Sør-Sverige + Danmark mørkt
  - Romerike mister strømmen etter "frisk bris"

HIO -Anvendt datateknologi - Kirsten Ribu 2007

7

## Avhengighet av IKT i kritiske samfunnsfunksjoner

- **Automatiserte datasystemer – det er sensorer overalt**
  - Overvåker prosesser i oljeproduksjon, strømforsyningen, RFID, biler etc
  - Erstatter mennesker
  - Offentlige og private tjenester på nettet – bank, selvangivelse osv
  - Samfunnskritiske systemer kobles til Internett

HIO -Anvendt datateknologi - Kirsten Ribu 2007

8

## Datasystemer i nettverk styrer kritiske funksjoner i samfunnet

- Energiproduksjon og distribusjon, bankvesenet, telesystemer,
- offentlige tjenester, industriell produksjon, transport,
- produksjon av råvarer, helsetjenester etc.
- Vanskelig å gå tilbake til reserveløsninger uten bruk av IKT

HIO -Anvendt datateknologi - Kirsten Ribu 2007

9



## Firefox-feil blottlegger filer Popupblokkering inneholder feil

- En feil i programvaren for å stoppe popup-vinduer i Firefox har fått påpekt en feil som gir angripere tilgang til filer på brukerens datamaskin.
- <http://itpro.no/art/10113.html>

HIO -Anvendt datateknologi - Kirsten Ribu 2007

11

## Apple retter kritiske iTunes-feil

- Firmaet bak den utbredte medieavspilleren iTunes har gitt ut en oppdatering som fikser flere alvorlige feil i iTunes og støtteprogrammet QuickTime.
- Flere av feilene er betegnet som kritiske av Apple og kan føre til at en fremmed får tilgang til maskinen dersom programmene ikke oppdateres.
- <http://www1.vg.no/teknologi/artikkel.php?artid=168974>

HIO -Anvendt datateknologi - Kirsten Ribu 2007

12

## 100% oppetid, en myte?

- Da Netcoms datasystemer **knelet 28. april** hadde en maskinvarefeil gjort at Netcom la over lasten på backup-systemet. En programvarefeil i backupsystemet gjorde at dette ikke klarte jobben.
- It-leverandøren Siemens hadde lovet en oppetid på 99,999 prosent og har nå brukt opp nedetid for flere år fremover.
- Nylig stoppet også Oslo Børs opp en time på grunn av en datafeil og i forrige uke **sørget en feil hos EDB** at flere nettbanker og minibanker var nede i en og en halv time.
- Alle tre er eksempler på kritiske datasystemer som i teorien ikke skal bli utligjengelig for brukerne. Men det skjer. Hos noen skjer det oftere enn andre. Man kan spørre seg om tilnærmet 100 prosent oppetid er en myte, om leverandørene lover noe de ikke kan holde og kundene kjøper noe de ikke kan få?

HIO -Anvendt datateknologi - Kirsten Ribu 2007

13

## Kjente systemfeil Det Norske Veritas rapporterer:

- **Programvare utgjør en stadig økende andel av sikkerhetssystemer** og de samlede kostnader om bord i rigger og skip. Det er dog av avgjørende betydning at denne programvaren er forberedt for å fungere sammen, at svikt ikke oppstår og at kritiske situasjoner kan unngås.
- Alvorlige hendelser som stans i boring og produksjon fra flytende oljeplattformer, sammenstøt mellom skip og skipsforlis kan være forårsaket av manglende samspill mellom kritiske datasystemer.

HIO -Anvendt datateknologi - Kirsten Ribu 2007

14

## Systemfeil – Det Norske Veritas

- Moderne, maritime maskinanlegg er konstruert ved å integrere ulike datastyrte kontrollsystemer og delsystemer.
- Disse systemene er ofte levert av **flere leverandører. Dette har resultert i økt kompleksitet**, som igjen ikke er blitt tilstrekkelig fulgt opp under utviklingen av kvalitetssikrings- og testopplegg.

HIO -Anvendt datateknologi - Kirsten Ribu 2007

15

## Livsfarlig situasjon i London

- En mann så en person falle overende, og prøvde å tilkalle ambulanse.
- Det kan være frustrerende å ha dårlig tid og drosjesentralens telefon bare svarer at "de står nå i kø, vennligst ikke legg på røret".
- Vedkommende prøvde å ringe fra flere telefoner samtidig, men etter fem minutter ga han opp fordi han ikke kom fram.

HIO -Anvendt datateknologi - Kirsten Ribu 2007

16

## Livsfarlig situasjon i London

- Dette skjedde i London - etter at ambulansetjenesten **hadde innført et nytt og meget ambisiøst datasystem.**
- I det aktuelle tilfellet var det heldigvis "bare" et epileptisk anfall, og vedkommende klarte seg.
- I et annet tragisk tilfelle døde en 11 år gammel jente som bodde to minutter fra et sykehus: Det tok 53 minutter før ambulansen dukket opp.

HIO -Anvendt datateknologi - Kirsten Ribu 2007

17

## Testmetoder mangler

- Dette fører til en situasjon hvor
  - **teknologiutviklingen ligger foran testmetodene**
- og
  - **et behov for omfattende testing av sammensatte maskinsystemer.**

HIO -Anvendt datateknologi - Kirsten Ribu 2007

18

## Systemfeil - klassifisering

- Problemer for **individer**
- Systemfeil som **påvirker mange brukere**
- Problemer i kritiske systemer der mennesker **kan skades eller miste livet**

## Evaluering

- **Evaluering underveis**
  - Undersøkelse av konsistens mellom de ulike beskrivelsene av systemet (UML-modeller, beskrivelser databasen, og prosjektbeskrivelsen)
  - En test av det kjørende systemet.
- **Ved overlevering**
  - Er kontrakten oppfylt?
  - Kan prosjektet erklæres avsluttet?

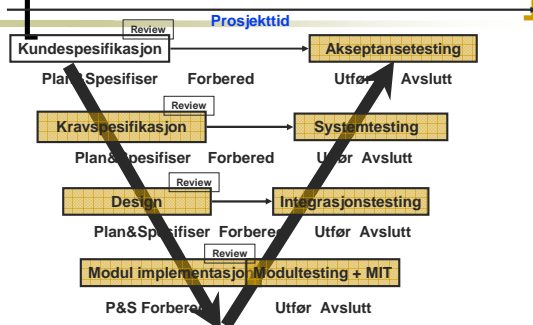
## Krav til test og verifikasjon

- **Risikoanalyse**
  - Hva er de verste risikomomentene med applikasjonen?
  - Bør egentlig gjøres i alle prosjekter
- Test skal konsentreres på **risikoområder**
  - Dersom systemet er løst koblet, kan en sortere ut områder med lavere risiko.

## Krav til test og verifikasjon

- Bestem et sikkerhetsnivå:
- Ulike standarder gir ulike nivåer. Typisk fra 0 til 4.
  - 0 = skader kan glemmes
  - 4 = truer mange menneskeliv etc.
- Nivå kan bestemmes pr. delsystem
- Nivå utløser krav om verifikasjon og test
- Nivå utløser ulik grundighet

## V-modellen



## Validering & Verifikasjon

- **Validering:** "Bygger vi det riktige systemet?"
  - Snakke med brukerne
  - Bruke use case modellen
- **Verifikasjon:** "Bygger vi systemet riktig?"
  - Manuelle inspeksjonsmetoder
  - Automatisert testing

## Hvorfor validering og verifisering?

- For å kunne vurdere hva vi gjør og hvorfor vi gjør det
- Behov for verifisering øker med størrelsen på systemet
- **Ca 1/3 av utviklingstiden brukes å testing**
- **Noen ganger opp til 50%**
- Systemutvikling er en **industriell prosess!**

HIO -Anvendt datateknologi - Kirsten Ribu 2007

25

## Statisk og dynamisk verifisering

- **Inspeksjoner** (statisk verifisering)
  - Analyse av systemet
  - kodeinspeksjon og gjennomgang av dokumentasjon for å oppdage problemer
- **Testing** (dynamic verifisering)
  - Observasjon av systemoppførelse
  - Systemet kjøres med testdata

HIO -Anvendt datateknologi - Kirsten Ribu 2007

26

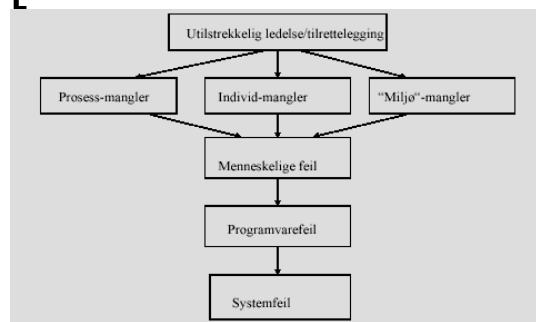
## Mennesker gjør feil

- Det vi lager er **ikke det vi burde laget**
- Det vi lager har **defekter**
- Vi trenger ikke nødvendigvis "best mulig" kvalitet: **Godt nok er bra nok.**
- **Jo senere en feil oppdages, desto mer alvorlig er det**
- Feil kan være **forretningskritisk** (i ytterste konsekvens kan menneskeliv gå tapt)

HIO -Anvendt datateknologi - Kirsten Ribu 2007

27

## Hva slags feil?



HIO -Anvendt datateknologi - Kirsten Ribu 2007

28

## Hvorfor finnes feil i ferdige produkter?

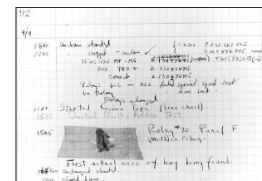
- **Det som er laget er feil (ikke det kunden vil ha)**
- **Ikke alle feil prioriteres rettet:**
  - Kategori 1: Kritiske feil som **MÅ** rettes (lansering holdes igjen til feilen er rettet)
  - Kategori 2: Kritiske feil som bør rettes (betydelig reduksjon av kvaliteten)
  - Kategori 3: Ikke-kritiske feil (kosmetiske feil)
- **Ikke alle feil finnes (tendens til å tro at feilen funnet sist er den absolutt siste feilen....)**

HIO -Anvendt datateknologi - Kirsten Ribu 2007

29

## Det første virkelige tilfelle av "bug"

9. september 1945, kl. 3:45 p.m., fant forskere ved Harvard universitetet årsaken til at Mark II Aiken Relay kalkulatoren oppførte seg merkelig  
En møll ble funnet fanget mellom punkter på relé #70, panel F  
Maskinen ble "debugget" med en pinsett!  
Dokumentert i loggen som "First actual case of bug being found." med møllen tapet inn ved siden av



HIO -Anvendt datateknologi - Kirsten Ribu 2007

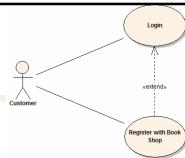
30

## Hvordan sikrer vi at programvaren er feilfri

## Feil i produktene

- **I praksis er det umulig å få verifisert alle kombinasjoner av input til et system**
- Det er en tendens til å teste og vektlegge bekreftelser, i motsetning til å prøve å falsifisere.
  - NB! Det er en vesentlig forskjell i holdning mellom det å utvikle og det å teste.
  - En god utvikler er "**konstruktiv**", mens en god tester er "**destruktiv**".
  - Mange organisasjoner **velger derfor å skille rollene, dvs. å ha egne testere.**

## Hvordan teste?



- **Finn ut om vi har laget systemet riktig:**
  - Sammenlign produktet med kravspesifikasjonen
  - Kravspek'en er visualisert i **use case modellen og beskrevet i use case'ne**

## Kvalitetssikringsstandarder

- **ISO 9000 – familien**
  - ISO 9001 standarden beskriver kvalitetssikringsarbeidet i hele livssyklusen
  - Utviklingsprosesser er også en kvalitetssikringsprosess

## Dokumentgjennomgang

- **Gjennomgang av dokumenter med formål å finne feil og mangler**
- **Forskjellige teknikker kan benyttes**
  - "parprogrammering" (kontinuerlig inspeksjon)
  - Forskjellige typer gjennomgangsmøter (dokumentet presenteres i møtet, distribueres på forhånd)

## Dokumentgjennomgang

- **Hvem bør gjennomgå dokumentet?**
  - De som skal ha systemet
  - De som skal bruke dokumentet
  - De som har vært delaktige i utformingen av dokumentet
  - Ekspertene (rollen / forretningsområdet)

## Kodegjennomgang

- **Funksjonelle feil** (avvik fra design og kravspesifikasjon)
- **Avvik fra kodenstandard** og retningslinjer (for eksempel navngiving av klasser)
- **Tekniske feil** (for eksempel feil i en metode)

## Kodegjennomgang

- **Forskjellige teknikker kan benyttes**
  - Parprogrammering (kontinuerlig kodegjennomgang)
  - Distribusjon til en eller flere for gjennomlesning
  - Koden gjennomgås av en review-gruppe i et møte

## Inspeksjons-team

- **Hvem bør gjennomgå kode?**
  - Sjefsprogrammerer/designere
  - Juniorprogrammerere (opplæring)
  - Testere
  - Tekniske eksperter
  - Arkitekter

## Inspeksjons-team

- **Minst 4 medlemmer i teamet:**
  - Den som har laget koden
  - Inspektør som finner feil, uoverenstemmelser, mangel på konsistens
  - Oppleser som leser koden for teamet
  - Møteleder som noterer feilene som blir funnet

## Fordeler

- **Mer enn 60%** av alle feil kan oppdages ved kodeinspeksjon
- Med matematisk verifisering kan **90% av feil i koden oppdages.**
- Mange ulike defekter oppdages under en enkelt inspeksjonsrund
  - **(NB! Ved testing: Kan ofte bare oppdage 1 feil - programmet kræsjer f.eks ved feil)**

## Foreler

- **Gjenbruk av kunnskap**
  - Inspektørene har sannsynligvis sett vanlig feil relatert til programmeringsspråket og i den type applikasjoner.
  - Derfor er det fokus på denne type feil under inspeksjonen

## Type test

- **Enhetstest/funksjonstest**
  - Hvem tester: Programmerer
- **Integrasjons- og systemtest**
  - Hvem tester: Tester
- **Akseptansetest**
  - Hvem tester: Installatør og kunde
- **Drift:**
  - Kunde

HIO -Anvendt datateknologi - Kirsten Ribu 2007

43

## Typer testing

- **Enhetstest**
  - Tester at komponenten (klassen) virker isolert
  - Simulerer omgivelsene til komponenten
  - Utføres av utviklere
  - Skrives ofte som automatiske tester
- **Integrasjonstest**
  - Tester at komponenten (klassen) virker sammen med andre komponenter
  - Simulerer ofte andre sub-systemer
  - Bruker konstruerte testdata
  - Utføres ofte av utviklere
  - Gjøres ofte manuelt, men kan med fordel automatiseres

HIO -Anvendt datateknologi - Kirsten Ribu 2007

44

## Testing

- At et system har "stor" utbredelse, er ingen garanti for at det er feilfritt.

HIO -Anvendt datateknologi - Kirsten Ribu 2007

45

## Typer testing forts.

- **Betatest**
  - Utvalgte kunder tar i bruk systemet før offisiell lansering
- **Akseptansetest**
  - Tester at systemet lar brukerne gjøre det de trenger
  - Tester med reelle data
  - Utføres gjerne i samarbeid mellom kunder og testere
- **Systemtest**
  - Tester at systemet oppfører seg korrekt i samspill med omgivelsene

HIO -Anvendt datateknologi - Kirsten Ribu 2007

46

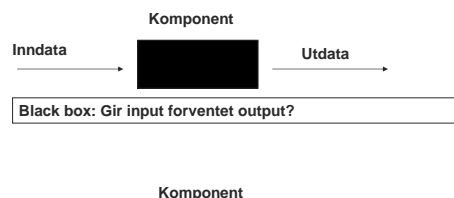
## Type testing forts.

- **Ytelsestest** (tester ytelse = hastigheten på én transaksjon)
- **Stresstest (overbelastningstest)** (tester skalerbarhet = hastigheten på mange samtidige transaksjoner)
- **Recoverability-test** (tester systemets håndtering av uforutsette avbrudd)

HIO -Anvendt datateknologi - Kirsten Ribu 2007

47

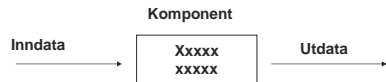
## Black-box testing



HIO -Anvendt datateknologi - Kirsten Ribu 2007

48

## White-box testing



White box: Følger hvert trinn i komponenten. Midlertidige utskrifter.

## Bruke use casene til testing

- Er pre- og postbetingelsene oppfylt?

## Risikoanalyse

- Systematisk fremgangsmåte for å beskrive og/eller beregne risiko.
- Risikoanalysen utføres ved kartlegging av uønskede hendelser, og årsaker til og konsekvenser av disse.
- I mange tilfeller vil det være tilstrekkelig å identifisere sannsynlige feil, slik at de nødvendige elementer kan valideres.

## Empiriske brukertester

- **Hypotese:** Vi har laget det riktige programmet
- La framtidige brukere anvende programmet til sine oppgaver
- 1. Observasjon:
  - Observér hvilke problemer de har og hvilke feil de gjør
- 2. Måling
  - Mål hvor lang tid de trenger for å lære programmet eller løse en oppgave
  - Tell antall feil, antall tastetrykk, ...
- 3. Intervju
  - Spør om det var dette de trengte eller ville ha
  - Brukernes subjektive oppfatninger

## Intervju

- Spørre brukere om hva de bruker systemet til og hvor mye de bruker det
- deres personlige oppfatning eller opplevelse av systemet
- Ikke nødvendigvis noen sammenheng med hvilke resultater som oppnås ved hjelp av systemet
- Brukere synes endringer er brysomme
- Brukere vil ha eksisterende funksjonalitet pluss litt til