Software Engineering (I00094)

College 4: *Ontwerp*

Marko van Eekelen
marko@cs.ru.nl
kamer HG02.074

Huidige planning

1. 6 feb: Het systeemontwikkelproces
2. 13 feb: Requirements-analyse
3. 6 mar: Documentatie, ProcesKwaliteit
4. Do 15 mar : Gastcollegereeks via Thalia Software Engineering Symposium
5. 27 mar : **Ontwerp**
6. ... : Menselijke factoren
7. ... : ...
8. ... : ...

Opgaven tot nu toe...

	Aia	Logi	Sosa	Tool
Klantgesprek	x	x	x	x
Requirements	x		x	
Kwaliteit		x	x	x
#mails	61	163	97	43

College 4: Leerdoelen

- Ontwerpconcepten en principes
- Kenmerken van een goed ontwerp
- Hoofdstuk 9 van Pressman

Ontwerp

- Definitie:
  Een representatie van
  iets dat gebouwd gaat worden,
  die terug te voeren is tot
  de requirements die eraan gesteld zijn

- 4 aandachtsgebieden
  - Data
  - Architectuur
  - Interfaces
  - Componenten

Het ontwerppproces

- **Diversificatie**

  het genereren van zoveel mogelijk alternatieve oplossingen
- **Convergentie**

  het selecteren van de meest geëigende oplossing in de gegeven context

Design principles

"The beginning of wisdom for a software engineer is to recognize the difference between getting a program to work and getting it right"

M.A. Jackson

Concepten

1. Abstractie
2. Stapsgewijze verfijning
3. Modulariteit
4. Structuur - Architectuur
5. Information hiding
6. Afhankelijkheid
7. Samenhang (Cohesie)
8. Koppeling

1. Abstractie

- Weglaten van detail
- Groepeer zaken van eenzelfde abstractieniveau
- Soorten abstractie
  - Procedurele abstractie
  - Data-abstractie
  - Controle-abstractie

2. Stapsgewijze verfijning

- Top-down ontwerpstrategie
- Laat je niet verleiden om in een vroeg stadium teveel details vast te leggen

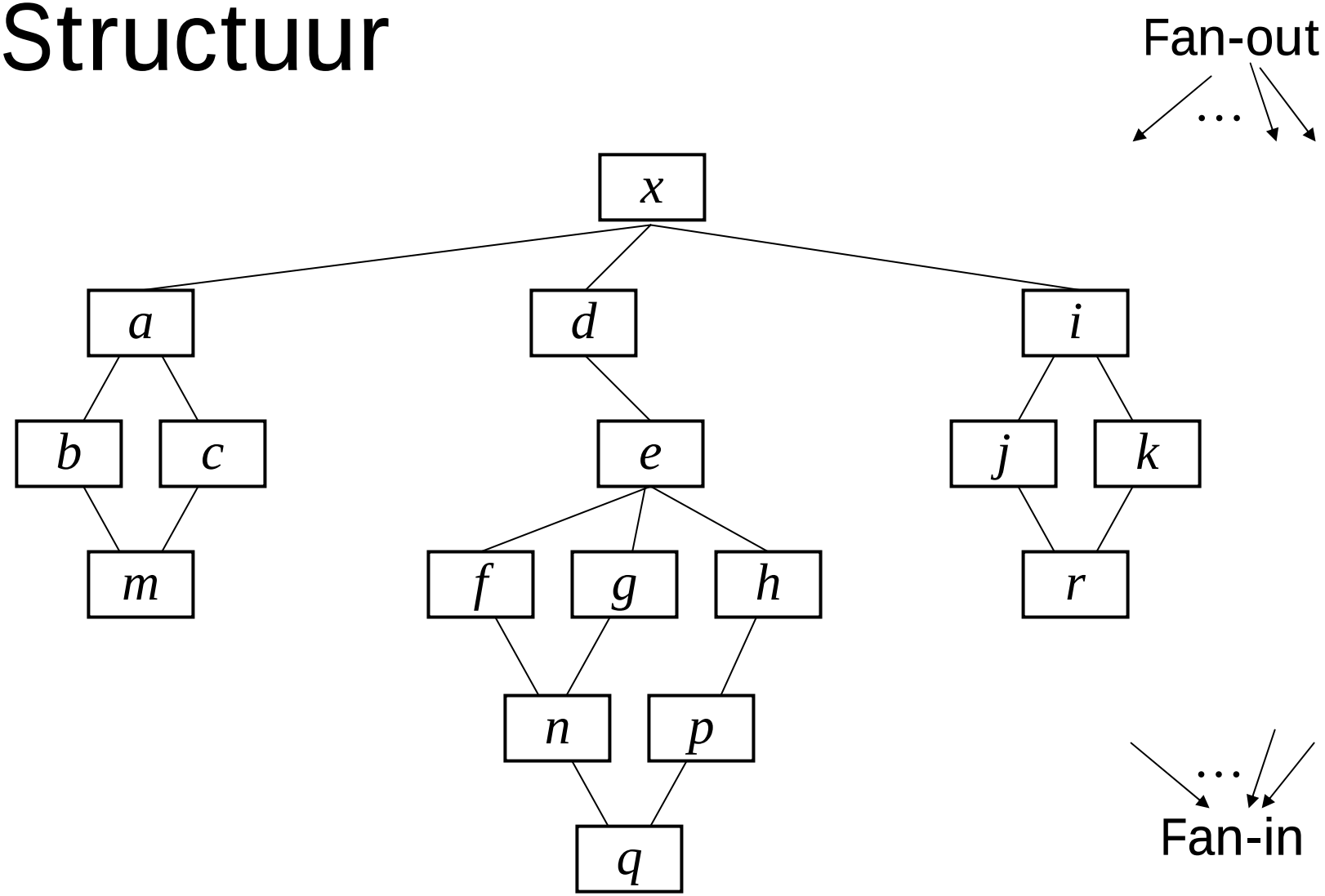
3. Modulariteit

- Divide and conquer
- Trade-off in complexiteit
  - modules vs integratie/communicatie
- Aandachtspunten
  - Decomposability
  - Composability
  - Understandability
  - Continuity
  - Protection

4. Structuur - Architectuur

- Componenten en de communicatie ertussen
- Niet-functionele eigenschappen
  - Structuur is vaak bepalend voor niet-functionele eigenschappen zoals performance, robuustheid en flexibiliteit
- Patterns, hergebruik

Structuur



5. Information hiding

- Zaken die onzichtbaar zijn voor de buitenwereld kunnen zonder risico gewijzigd worden
- Voorbeelden
  - Private variabelen/methoden
  - Inner classes

6. (On)afhankelijkheid

- Ontwerp modules met
  - Eenduidige functie
  - Aversie voor communicatie
  - Simpele interfaces
- Onafhankelijkheid wordt bepaald door
  - Cohesie
  - Koppeling

7. Cohesie

- Samenhang tussen delen van een module
  - Logisch
  - Temporeel

8. Koppeling

- Streef naar zo laag mogelijke koppeling
- Voorbeelden:
  - Aanroepen
  - Globale variabelen
  - Filesysteem
  - Overerving

Ontwerpheuristieken

- Hergroepeer modulen nadat een eerste opzet van de structuur gemaakt is
- Vermijd hoge fan-out
- Bevorder fan-in voor low-level procedures
- Scope of effect $\subseteq$ Scope of control
- Simpele interfaces
- Voorspelbare modules
- Vermijd (arbitrare) restricties
- Controlled entry

Ontwerp van gebruikersinterfaces

A. Zet de gebruiker aan het stuur
B. Beperk de memory-load voor de gebruiker
C. Maak een consistente interface

A. Gebruiker aan het stuur

- Geen onnodige acties
- Flexibiliteit
- Onderbreekbaar, undoable
- Verberg technische details
- Direct manipulation

B. Memory-load

- Laat het systeem zoveel mogelijk informatie onthouden die later opnieuw gebruikt zou kunnen worden
- Defaults
- Gebruik metaforen
  - Spatiële metafoor
  - Sensor-motor metafoor
- Presenteer niet teveel informatie in eens

C. Consistentie in user interfaces

- Maak functies en contexten herkenbaar
- Beperk het aantal verschillende soorten acties
- Documenteer je ontwerpstijl, en hou je eraan
- Kijk naar het type applicaties dat de gebruiker al kent

Geen nieuwe opgaven meer

- You are on your own now.....
- Maar wel snel de oude inleveren....