KULeuven

3 Ba Informatica

Beleidsinformatiesystemen

**Beleidsinformatiesystemen: Lesnota’s**

*Author:*

Helena Brekalo

May 22, 2015

# Contents

1. **Les1: 09/02/2015 4**
   1. Slides: 1a. What is an IS . . . . . . . . . . . . . . . . . . . . . . 4
      1. Deel 1: Motivation . . . . . . . . . . . . . . . . . . . . . . 4
      2. Deel 2: What is an IS? . . . . . . . . . . . . . . . . . . . . 7
2. **Les 2: 13/02/2015 9**
   1. Slides: 1a. What is an IS . . . . . . . . . . . . . . . . . . . . . . 9
      1. Deel 2: What is an IS? . . . . . . . . . . . . . . . . . . . . 9
      2. Deel 3: Types of information systems . . . . . . . . . . . 10
   2. Slides: 1b.ISstrategy . . . . . . . . . . . . . . . . . . . . . . . . . 12
      1. Deel 1: Introduction . . . . . . . . . . . . . . . . . . . . . 12
      2. Deel 2: Porter’s competitive forces model . . . . . . . . . 12
3. **Les 3: 16/02/2015 15**
   1. Slides: 1b.ISstrategy . . . . . . . . . . . . . . . . . . . . . . . . . 15
      1. Deel 3: Competitive advantage and IS: the value chain

perspective . . . . . . . . . . . . . . . . . . . . . . . . . . 15

* + 1. Deel 4: Linking business strategy and IS strategy: Hen-

derson & Venkatraman’s strategic alignment model . . . . 18

1. **Les 4: 20/02/2015 21**
   1. Slides: 2A. Creating Value with IT . . . . . . . . . . . . . . . . . 21
      1. Business Value of IT . . . . . . . . . . . . . . . . . . . . . 21
      2. If we build it, will they come? . . . . . . . . . . . . . . . . 23
2. **Les 5: 23/02/2015 24**
   1. Slides: 2A. Creating Value with IT . . . . . . . . . . . . . . . . . 24
      1. If we build it, will they come? . . . . . . . . . . . . . . . . 24
      2. Information System Success . . . . . . . . . . . . . . . . . 26
      3. Case Study . . . . . . . . . . . . . . . . . . . . . . . . . . 27
3. **Les 6: 27/02/2015 32**
   1. Pop quiz . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32
   2. Slides: 2B. EnterpriseArchitecture . . . . . . . . . . . . . . . . . 33
      1. Why? . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33
      2. What? . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 36
      3. How? . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37
4. **Les 7: 02/03/2015 41**
   1. Slides: 2B. EnterpriseArchitecture . . . . . . . . . . . . . . . . . 41
   2. Slides: 3A BPManagementIntro . . . . . . . . . . . . . . . . . . . 46
5. **Les 8: 06/03/2015 51**
   1. Slides: 3B BusinessProcessModellingEssentials . . . . . . . . . . 51
   2. Slides: 3C-BPMNBasics . . . . . . . . . . . . . . . . . . . . . . . 55
6. **Les 9: 09/03/2015 59**
   1. Slides: 3E BPEnactment . . . . . . . . . . . . . . . . . . . . . . . 59
7. **BPMN AM Activities** v **Les 10 65**
   1. Slides: 3D BPMN Advanced . . . . . . . . . . . . . . . . . . . . 65
      1. Video: BPMN AdvancedModeling: Activities . . . . . . . 65
      2. Video: BPMN Advanced Modeling: Events . . . . . . . . 67
      3. Video: BPMN Advanced Modeling: Modeling Alternatives 70
8. **Huistaak: BPMN Quiz 73**
   1. Slides: BPMN Quiz . . . . . . . . . . . . . . . . . . . . . . . . . 73
9. **Les 11: 16/03/2015 77**
   1. Slides: 4ABSummary . . . . . . . . . . . . . . . . . . . . . . . . . 77
10. **ER Model** v**Les 12 84**
    1. Slides: 4A Information Modeling ER Modelling . . . . . . . . . 84
       1. Video: Business Information Systems 31: Information

Management: ER Model [Dutch] . . . . . . . . . . . . . . 84

* 1. Slides: 4A Information Modeling ER Modelling . . . . . . . . . 91
     1. Video: Business Information Systems 32: Information

Management: EER Model [Dutch] . . . . . . . . . . . . . 91

* 1. Slides:4B Information Modeling RelationalModel . . . . . . . . . 94
     1. Video: Business Information Systems 33: Information

Management: Relational Model [Dutch] . . . . . . . . . . 94

1. **Huistaak** v**Les 12 100**
   1. Slides: Questions EER Normalisation StudentVersion(1) . . . . . 100
2. **Les 13: 23/03/2015 104**
   1. Slides: 4D SQL . . . . . . . . . . . . . . . . . . . . . . . . . . . . 104
3. **Huistaak** v**Les 14 111**
   1. Slides: 4C SQL . . . . . . . . . . . . . . . . . . . . . . . . . . . . 111
      1. Video: Business Information Systems 4: Structured Query

Language (SQL) . . . . . . . . . . . . . . . . . . . . . . . 111

1. **Huistaak: Oefeningen SQL 123**
   1. Word document: Questions SQL StudentVersion . . . . . . . . . 123
2. **Les 15: 30/03/2015 128**
   1. Slides: 1cITGovernance . . . . . . . . . . . . . . . . . . . . . . . 128
   2. Slides: 5. CustomerRelationShipManagement . . . . . . . . . . . 132
3. **Les 16: 03/04/2015 135**
   1. Slides: 5. CustomerRelationshipManagement . . . . . . . . . . . 135
4. **Les 17: 27/04/2015 143**
   1. Slides: 6A.BusinessIntelligenceandDataAnalytics . . . . . . . . . 143
5. **Les 18: 04/05/2015 153**
   1. Slides: 6B.BusinessIntelligenceandDataAnalytics-studentversion . 153
6. **Les 19: 08/05/2015 161**
   1. Slides: 6B.BusinessIntelligenceandDataAnalytics-studentversion . 161
   2. Slides: 6C.BusinessIntelligenceandDataAnalytics-student version 166
7. **Lecture: Business Information Systems 9-1: E-business** v **Les**

**20 169**

* 1. Slides: 7A.E business part1 . . . . . . . . . . . . . . . . . . . . . 169
     1. Video: Business Information Systems 9-1: E-business . . . 169

1. **Les 20: 11/05/2015 179**
   1. Slides: 6C.BusinessIntelligenceandDataAnalytics-student version 179
   2. Slides: 7D.E-business-part1-QuizSession-Student Version . . . . . 183
2. **Lecture: Business Information Systems 9-2: E-business** v**Les**

**21 185**

* 1. Slides: 7B. E business part2 . . . . . . . . . . . . . . . . . . . . . 185
     1. Video: Business Information Systems 9-2: E-business . . . 185

1. **Les 21: 19/05/2015 198**
   1. Slides: 7D.E-business-part1-QuizSession-Student Version . . . . . 198
   2. Slides: 7E.E-business-part2-QuizSession-StudentVersion . . . . . 200
2. **Lecture: Business Information Systems 9-3: E-business** v**Les**

**22 204**

* 1. Slides: 7C.E-business-part3 . . . . . . . . . . . . . . . . . . . . . 204
     1. Video: Business Information Systems 9-3: E-business . . . 204

1. **Les 22: 22/05/2015 212**
   1. Slides: 7F.E-business-part3-QuizSession-StudentVersion . . . . . 212
   2. Slides: WrapUp . . . . . . . . . . . . . . . . . . . . . . . . . . . . 215

**Chapter 1**

# Les1: 09/02/2015

## 1.1 Slides: 1a. What is an IS

### 1.1.1 Deel 1: Motivation

**Slide 4:** Main questions you should be able to answer after watching this movie:

1. What are the challenges that today’s organizations are confronted with due to the technology explosion?
2. What are the things companies need to do to hit their digital sweet spot?
3. • Who’s accountable for digital? Some companies have created a chief digital officer or deriving the digital agenda from the IT function. But the most successful companies are having their chief executives involved.
   * How to match the speed and agility of organisations that are born digital; start-ups tend to move at a very different rate.
   * How to build digital skills across the enterprise? In order to compete effectively, businesses need digital skills; not just in marketing and sales but increasingly in operations and across the whole value chain.
4. • They need to understand really where is their value of digital: in marketing, in sales, automating operations, or a combination of all of those?
   * They need to prioritize: there are always too many things to do in the digital portfolio and focusing on the ones that count is important.
   * They need to take and end-to-end view: ensuring that customers receive a joined-up experience, end-to-end, and that all functions are working together → most challenging.
   * They need to look at their portfolio of businesses and understand what impact digital may have on evaluations (?) but also on the needed capabilities going forward and perhaps rebalance the portfolio accordingly.

Commentaar bij video:

**Vraag 1:** • Laat de strategie van het bedrijf niet afhangen van de mensen die alleen met IT bezig zijn (who’s accountable?). Vaak zien we dat het belangrijk is om 1 belangrijke persoon meer te hebben: CIO (Chief Information Officer).

* + Omgaan met explosie aan vaardigheden die nodig zijn. Je moet op zoek gaan naar de juiste mensen om met IT binnen het bedrijf te kunnen omgaan. Het is niet vanzelfsprekend om mensen in huis te hebben die met de trends meegaan.
  + Overzicht over IT: het hebben van de juiste capaciteiten en overzicht hebben op over welke gebieden IT een invloed heeft. IT inzetten in alle functionele domeinen waar het het meest kan opbrengen.

**Vraag 2:** • Kijk naar het portfolio: end-to-end verhaal: ze moeten een

portfolio-view hebben, kijken naar de verschillende functionele domeinen. Kijken welke domeinen interessant kunnen zijn. Kijken hoe die aan elkaar geschakeld zijn.

* + Wat zijn de prioriteiten? Zijn altijd gelinkt aan budgetten, geld dat je kan investeren. Je kan niet zomaar zeggen dat je plots alles digitaal gaat doen. Volledige herorganisatie van IT gaat niet zomaar. Welke investeringen/technologieen kunnen waarde opleveren voor de klanten en zo de winst omhoog helpen?
  + De waarde van IT kennen → is zeer moeilijk, niet elk bedrijf kan dat zomaar. Je moet inzicht hebben in hoe je met IT extra waarde kan cre¨eren voor je klanten.

**Slide 6 en verder:** IT heeft een enorme impact op ons leven gehad (trein te laat: via smartphone checken). Als we kijken hoe groot die impact is voor organisaties: groter dan die op ons persoonlijk leven?

**Slide 11:** App om straten te laten fixen: gigantische impact door 1 simpele app. Inzicht in fout in infrastructuur in de stad wordt gigantisch vermendigvuldigd. Onderhoud van wegen etc. is niet gedigitaliseerd, maar de app zorgt voor verandering van aanpak (→ veel meer mensen nodig want veel meer meldingen).

**Slide 12:** Als we kijken naar de realisatie van het inzetten van IT in een organisatie: kijken naar de software-implementatie geeft gegeven pie-chart:

* + 18% van de projecten falen gewoon. Elke cent is verloren.
  + 53% zijn challenged: bij de start van de request/requirements had je een lijstje; op het moment dat het systeem wordt opgeleverd, zijn er requirements die niet voldaan worden (bv voor straat: moet in real-time werken; bij aanmelden: defecten doorsturen naar server, dat kan bv falen, slechts 1/3 komt correct toe wegens verschillende mogelijke redenen).
  + 29% is succesvol.

Standish group gaat op zoek naar waarom zoveel dingen foutlopen. Main reason: business process ignorance: het gaat erom dat het al fout loopt bij de requirements (opstellen ervan), op strategisch niveau (wordt enkel overgelaten aan IT om te ontwikkelen, geen mensen met business-kennis); je denkt niet na over welke stappen ondernomen moeten worden. Je moet duidelijk specifi¨eren waaraan de app moet voldoen, als je geen inzicht daarin hebt, loopt het vaak fout. Geen kennis van het bedrijfsaspect van hoe je waarde levert aan je klant. ⇒ Business-IT alignment.

**Slide 14:** Rapport van de Standish Group

**Slide 15:** Omgekeerde van slide 14 → met klanten overleggen (zeker als je zelf nog niet weet wat er moet gebeuren), klanten betrekken (user involvement). Clear vision: de belangrijkste mensen in de organisatie moeten bepaalde ITinvestering ondersteunen. Belangrijke mensen in de organisatie moeten erachter staan.

**Slide 16:** Impact van IT: als die er is, loopt het vaak fout. Slide 16 zegt hoeveel waarde je kan cre¨eren met IT. De conclusie is dat minder dan 10% van het budget zal bijdragen tot hogere waarde (klanttevredenheid,...). 75% gaat naar enkel onderhoud en het runnen van systemen. 25% gaat naar nieuwe projecten en daarin komt de verhouding tussen sucessful, challenge en failed terug. 2/3 daarvan is challenged/failed. 1/3 van die 25% leidt tot nieuwe, toegevoegde waarde door IT. Flagrant laag! Dus, hoe groot is de impact van IT? Want als we aan dit rendement komen, lijkt het niet zo denderend.

**Slide 17:** Effectieve impact van IT op organisaties: investeringen lijken tot niet veel te leiden aan de hand van de vorige slide. We willen specifieker gaan kijken: macro- en micro-economische impact.

**Slide 18 ev:** Friedman schreef de tekst ”It’s a flat world ater all”: zijn visie van IT op de wereld, bespreekt de impact van IT. Geeft een heel goed inzicht in hoe hard IT macro-economisch gezien de wereld heeft veranderd. Er zijn 10 verschillende flatteners die zorgen dat de wereld 1 level-plain field is. Tussen bedrijven wereldwijd zijn barri`eres weggenomen door IT (vooral concurrentie). Technologie en internet hebben de barri`eres weggenomen. De rode flatteners zijn de belangrijksten. Concurrentie wordt op een andere manier ingevuld, veel geglobaliseerder. Steroids: pure technologie: robottechnologie, hoe chips effici¨enter gemaakt kunnen worden,... Die evolutie heeft ook al een gigantische impact op ondernemingen. Onze smartphone nu is performanter dan de computer van 10 jaar geleden. Pure technologie verbetert steeds → bedrijven concurreren anders. De globaliserende wereld globaliseert veel sneller door technologie. De grootste bedrijven zijn er gekomen door die impact (Apple, Google, Facebook,...) → op macro-niveau heeft IT een gigantische impact. De vraag blijft waarom het zo vaak fout loopt. (**Slide 20**) → Op micro-economisch gebied: bedrijven die falen op IT, gigantisch grote fouten maken, moeten opgedoekt worden omdat ze niet op de juiste manier met IT kunnen omgaan. Op micro-economisch vlak maakt IT een verschil voor de organisatie op zich.

**Slide 21:** Quotes die zeggen dat IT niet belangrijk is, maar eerder tegenwerkt. Productiviteit van organisaties: is gaan kijken naar moment waarop computers gebruikt begonnen worden en keek wat de impact op de productiviteit was. Hij stelde dat de productiviteit er net door daalde, niet steeg.

**Slide 22:** Nicholas Carr: IT is een utility zoals water of energie (pure verbruiksgoederen). Hij probeerde aan te tonen dat IT een gelijkaardige curve vertoont als bv railways, electric power, water,... IT is om te verbruiken, maar innoveren in IT is niet nodig. Eens er spoorwegen waren, werd er niet meer zoveel ge¨ınnoveerd op dat vlak. IT is een nutsvoorziening, net zoals treinsporen. Hij zegt dat je moet volgen, niet leiden. Wachten tot er door anderen stappen gezet worden. Zodra er zo successen behaald worden, kan je ook die aankoop doen. Pas investeringen doen als het mainstream wordt. Carr’s opinie is dat als je nog niet gezien hebt dat iets (extra) waarde kan opleveren, je dan nog niet moet volgen. Nooit als eerste bedrijf/organisatie kijken of een IT-innovatie ons iets kan opbrengen. Je moet zorgen dat er geen gaten ontstaan tussen wat mainstream is en wat jij doet. Zorg dat je blijft volgen, maar zelf niet kijken naar nieuwe investeringen, nieuwe zaken die interessant kunnen zijn.

**Slide 25:** Niet akkoord: Carr ziet IT puur en alleen als infrastructuur, als zijnde het serverpark, PC’s,... → De assets. Dat is natuurlijk niet het enige. Bij IT gaat het niet alleen om servers,... die in de digitale wereldbeschikbaar moeten zijn. Er zit veel meer achter. Het zit in de technologie *en* de mensen: nadenken wat je met de technologie gaat doen. Het gaat niet alleen om de juiste assets, maar ook nadenken over hoe de IT de zaken kan ondersteunen, de mensen die proberen na te denken over hoe het nieuwe waarde kan cre¨eren. ⇒ Manier waarop je IT gebruikt. Solow: het is niet makkelijk om productiviteit te meten. Solow keek naar de groei van economie, dat kan kloppen, maar je moet ook gaan kijken naar andere zaken die moeilijker te meten zijn zoals bv kwaliteit van de producten. Het is niet eenvoudig om productiviteit te gaan meten. Er is een delay waarop IT een impact gaat hebben, kan long-term zijn: een *lag*. Het kan een tijd duren na de investering vooraleer je vooruitgang ziet. De zaken moeten stabiliseren. Wachten tot stabiele technologie-omgeving. Technologie verandert constant, daarom is het niet makkelijk om de productiviteitsveranderingen te meten. Het gaat niet enkel om automatisering, maar ook om het bruikbaar maken van automatisering/digitalisering. Je moet ook je werknemers meekrijgen. Conclusie: IT heeft effectief impact, zowel op macroals op micro-economisch vlak. Als je mensen en technologie samenbrengt, kan je waarde cre¨eren. Het gaat niet om het vervangen van het denken, maar hoe je complementair kan zijn, beter kan denken.

**Slide 28:** We kunnen niet zomaar zeggen dat IT bij de IT people hoort en wij business mensen kijken er niet naar.

**1.1.2 Deel 2: What is an IS?**

Wat is informatie? Belangrijk verschil tussen data en informatie: **slide 31** toont geen informatie omdat je een extra stap nodig hebt: je moet het eerst gaan verwerken. Zonder een computer is die slide data, heeft het geen betekenis, is niet relevant. ⇒ Het grote verschil tussen data en info: data gaat over ruwe feiten, maar het gaat over zaken die we niet kunnen interpreteren. Symbolische cijfers, getallen, documenten,... zonder betekenis, je kan er geen betekenis aan hechten. Zodra je dat wel kan, noemen we data informatie. Zodra je kan interpereteren wat er staat, wordt het informatie. Wat heb je hiervoor nodig? Context. Dus: *data* + *context* = *informatie*. Afhankelijk van de individuele context. Kennis: van informatie naar kennis: je kan niet alleen begrijpen wat er staat (info), maar je kan ook actie ondernemen, zeggen wat er moet gebeuren. Je kan beslissingen nemen aan de hand van de info.

**Slide 36:** Metadata: context die ervoor zorgt dat data informatie wordt. Informatie die data beschrijft en zo wordt data informatie (eigenlijk metainformatie). De context die ge¨ınterpreteerd wordt.

**Slide 37:** Soms heb je informatie nodig om de info relevant te kunnen maken (Dublin core). Bv als je niet weet wanneer een document de laatste keer gewijzigd is, ben je er soms niks mee.

**Slide 39:** Data wordt informatie door metadata.

**Chapter 2**

# Les 2: 13/02/2015

## 2.1 Slides: 1a. What is an IS

**2.1.1 Deel 2: What is an IS?**

**Slide 40 ev:** Wat is een systeem? Bestaat uit elementen, relaties (die elementen zijn gerelateerd aan elkaar, passen in een soort van universal discourse, een omgeving), heeft een bepaald doel voor ogen.

Point van slides erna: je kan het begrip ”systeem” vanuit verschillende perspectieven zien. Er zijn verschillende modellen/perspectieven om een systeem te beschrijven/zien.

**Slide 45:** Wij maken gebruik van een componentmodel. Een component hiervan: een proces. Ook omgeving is nodig, dat is de universal discourse in onze definitie. We hebben die nodig om het componentmodel correct te kunnen opbouwen. Ook interacties met omgeving nodig: input & output. De drie (gele) componenten en de environment maken een eerste componentsysteem. Ook managementcomponent nodig (staat in slides van prof op plaats van oranje rechthoek vanboven).

**Slide 48:** We kunnen de componenten vernauwen tot een information system. Het doel van ons systeem is het maken van beslissingen in zijn algemeenheid. We willen de informatie kunnen gebruiken om beslissingen te nemen, zij het tactisch, strategisch,...

**Slide 49:** Als we informatie beschouwen als het startpunt van ons componentmodel, dan zien we dat ruwe data (kan ook al informatie zijn) de input vormt, die verwerken (zoeken, verwerken, opslaan, distribueren, aggregeren) en we krijgen als output relevante informatie. De environment noemen we de managementcomponent. Als uw data niet correct gemodeleerd is, gaan we zien dat er fouten zijn rond beschikbaarheid van data,...

**Slide 50:** BIS kan gezien worden als een soort kopie (carbon copy, potloodkopie) van de echte organisatie, van het business systeem. In de utopische case wordt business perfect weerspiegeld in het informatiesysteem. Zou inhouden dat elke taak die onderneming doet zichtbaar zal zijn in informatiesysteem.

### 2.1.2 Deel 3: Types of information systems

**Slide 52:** Een model om soorten informatiesystemen te onderkennen. Eerste dimensie: horizonale dimensie: functional area, functions. Typisch hebben we informatiesystemen die bepaalde functies van een bedrijf ondersteunen (bv sales en marketing, productie,...). → Silos of information systems: afhankelijke blokken van het informatiesysteem, we zien relatief onafhankelijke blokken die onze informatiesystemen ondersteunen. Stel dat we die silos niet zouden hebben, idealiter zijn die verweven omdat ze invloed hebben op elkaar en hebben elkaars info nodig.

Kind of information system: operational: er zijn informatiesystemen die enkel dienen om processen te ondersteunen (bank: wordt volledig ondersteund door infosysteem, volledig op operationeel niveau), management en strategic zijn moeilijker van elkaar te onderscheiden. Ze dienen om het operationele niveau zo goed mogelijk te besturen.

Management: tactisch zodat operationeel zo effici¨ent en goed mogelijk uitgevoerd wordt. Strategic: board of directors etc. die beslissingen nemen, op het hoogste niveau, ook daarvoor bestaan specifieke systemen.

**Side 53:** Operationeel: dagdagelijkse operaties die uitgevoerd worden. Beslissen hoe het model er moet uitzien om te bepalen of iemand bv een lening kan krijgen = tactisch: welke elementen neem ik in rekening om te beslissen of iemand een lening mag aangaan (ga ik rekening houden met het loon, de leeftijd, het geslacht,...?). Welk soort (beslissings)model: een generieker model dan operationeel. Strategisch: langetermijnbeslissingen op niveau van de organisatie.

**Slide 54:** Verschillende doelen waarvoor de driehoek gebruikt kan worden.

**Slide 55 ev:** Rechterkant van de piramide: we hebben verschillende soorten informatiesystemen die de rollen in een systeem ondersteunen. Operationeel niveau: beslissingen die onderhevig zijn aan weinig onduidelijkheid, heel makkelijke beslissingen. De informatie die je nodig hebt is vaak heel makkelijk beschikbaar, heel gestructureerd, makkelijk te raadplegen. Systemen die we daarin zien zijn OLTP en ERP.

Voorbeeld OLTP: payroll systeem dat input neemt: payroll master file: neemt info om te bepalen hoeveel werknemers betaald moeten worden en een systeem zorgt ervoor dat betaling ook effectief wordt uitgevoerd.

**Slide 60:** ERP valt ook onder OLTP, is ook generieke term. ERP is speciaal: kan je visueel weergeven zoals op slide 60, tracht te realiseren om de verschillende aparte infosystemen (silo’s) samen te brengen. Ondersteuenen alle functionele domeinen en zorgen voor een groter niveau van integratie. Het productiesysteem is vaak totaal niet gerelateerd aan accountancysysteem bv, maar ERP brengt dat samen. Zorgt er inherent voor dat je meer integratie hebt en meer dataexchange tussen uw verschillende domeinen. Zorgt dat sales & marketing bij bv finance & accounting info kan opvragen.

Off-the-shelf/plain vanilla: typische ERP-vendors verkopen die soorten systemen, generieke implementaties: de generieke implementatie wordt overgenomen van de vendor, je past zelf niets aan aan dat ERP-systeem. Dus geen custom implementatie. Alternatief: wel customizen aan uw specifieke context. KULeuven doet dat zo: SAP-experten die het systeem aanpassen aan eigen noden.

**Slide 61:** Tactisch/management niveau: doel: ondersteunen van de mid-term beslissingen die veel minder routineus zijn, veel minder makkelijk zijn en minder info vereisen. Ze vereisen info van het operationeel niveau die we opslagen in data warehouses.

Verschil tussen MIS & DSS: MIS slaagt effectief op het controleren, sturen van het operationele niveau. Dat is het doel van MIS. Hoe gaan we dat doen? Input: transacties, operationele data, opslaan in data warehouses (& combineren) en simpele rapporten laten runnen, statistieken produceren om te controleren wat op het operationele niveau gebeurt. Typisch breed: controleren van operationele processen maar zijn qua output relatief simpel. Typische rapportjes, dashboards, samenvattingen, aggregaties van data.

**Slide 63** In rechthoek: data warehousesysteem. Rapporten kunnen statisch of real-time gecre¨eerd worden. In essentie geven ze wel dezelfde soort info, output, geen specifieke output. → Belangrijkste verschil met DSS!

DSS: gelijkaardige data als input (uit operationeel niveau samengebracht in data warehouse). We kijken niet naar alle data, naar alle processen, maar naar low volume en we willen hiermee specifieke vragen beantwoorden (bv data analytics). Geen generiek overzicht, je gaat specifieke analyses doen. Je gaat bv de vraag beantwoorden ”Welk model gebruiken we om kredietverlening te voorzien?”. Simulaties runnen om bv te zien hoeveel werknemers je nodig hebt voor een nieuw project.

**Slide 67:** Bij voyage-estimating DSS gaat het distributiesysteem correcte beslissingen laten nemen, houdt rekening met vervoerskosten etc. Gegeven alle inputs van prijzen voor allerlei zaken gaat die beslissingen nemen.

DSS ondersteunt specifieke vragen, MIS controleert en stuurt in meer algemene manier het operationele niveau.

**Slide 69:** SIS: om strategische beslissingen te kunnen nemen. Waar het risico groot is om foute keuzes te maken, veel complexere beslissingen. De info daar is veel moeilijker te vinden en verzamelen, heel sterk afhankelijk van de persoon die de data moet interpreteren. Mensen die strategische beslissingen nemen hebben vaak een heel specifieke kijk daarop.

Heel belangrijk: in tegenstelling tot MIS en DSS: gebruik van externe data! Bv info over marktomgeving (prijszetting in de sector, beurskoersen,...).

**Slide 70** (hoort bij slide 69): antwoorden op specifieke vragen.

**Slide 72:** Lijkt sterk op het management information system (van hierboven).

**Slide 73:** Transactiesystemen leveren data aan MIS en DSS, waarbij MIS vaak ook info levert aan DSS. Kunnen gebruikt worden om specifieke queries te beantwoorden.

**Slide 74 ev:** Piramide is niet perfect om informatiesystemen in bedrijven te classificeren. Er zijn systemen in bedrijven die je heel moeilijk kan positioneren: office automation systems en knowledge work. Office automation systems worden op operationeel niveau, management niveau,... gebruikt. → Situeren zich overal in de piramide (is er nu maar tussengezet als knowledge level, maar hoort zowat overal thuis). Zijn zeer populair, worden zeer frequent gebruikt in bedrijven. Zijn moeilijker te situeren, vaak relatief onafhankelijk van andere informatiesystemen. Specifieke systemen die kenniswerknemers (taak is om nieuwe kennis te produceren: bedrijven die onderzoeksafdeling hebben) ondersteunen:

KWS.

## 2.2 Slides: 1b.ISstrategy

### 2.2.1 Deel 1: Introduction

**Slide 3:** Begin met het oplijsten wat we van een CRM-systeem verwachten (24/7 bereikbaarheid? → Wat hebben we nodig om dat te ondersteunen?). Requirements komen vanuit verschillende perspectieven. Klanten bepalen steeds meer de ervaring op websystemen, klantervaring wordt steeds meer geanalyseerd om klanten meer te kunnen laten kopen etc. Customer requirements: (impliciete) eisen die klanten stellen op hun (klanten)ervaring spelen al een eerste belangrijke rol in het specifi¨eren van requirements.

User requirements: hoe moet het systeem eruit zien voor de gebruikers van het systeem?

Company goals: jouw bedrijf heeft een doel op zich. Spelen ook een rol om requirements te bouwen van een informatiesysteem (lage kostenstructuur om lage prijzen te kunnen garanderen?). Requirements zijn de enige en belangrijkste invloed om een analyse uit te voeren om te zien hoe uw systeem eruit moet zien.

**Slide 4 & 5:** Loopt heel vaak fout (miscommunicatie!).

**Slide 6:** De relatie tussen een informatiesysteem en de company goals (strategische doelen van een bedrijf).

### 2.2.2 Deel 2: Porter’s competitive forces model

**Slide 9 ev:** Porter’s competitive forces (zie OBT): basismodel in de strategie. Hoe is Porter aan dat model gekomen? Is een econoom die in de jaren ’80 heeft nagedacht over competitive environment, onder andere in de Amerikaans economie en vond de cijfers in slide 8: heel grote variabiliteit op winstgevendheid van sectoren. Airlines bv *<*6% profitability; soft drinks: bijna 40%. Porter wou dit verklaren, hoe kan het dat bepaalde sectoren zo winstgevend zijn en anderen niet? Klassieke manier om te kijken naar concurrentie schiet tekort (directe concurrentie tussen bedrijven die dezelfde producten leveren, bv Delhaize en Colruyt), Porter vond dat die manier van kijken onvoldoende was. Het feit ofdat je een heel sterke, hoog oplopende concurrentie hebt binnen een sector is onvoldoende om die grote verschillen in winstgevendheid te verklaren. Er moeten andere krachten zijn die het be¨ınvloeden.

* De middenste kracht (cirkel) is de klassieke manier om te kijken naar concurrentie (zie hierboven): prijsafspraken, kwaliteitscontrole,... → Interne competitie in de markt.
* Threat of new entrance: de dreiging dat er nieuwe bedrijven zullen toetreden tot de markt. Sommige markten zijn veel makkelijker toegankelijk dan andere markten (bv pizza restaurant vs chip maker; pizza: bijna niks nodig, je kan heel makkelijk tot die markt toetreden. Chip maker is niet van de ene dag op de andere op te starten: gigantisch veel kapitaal, kennis, geschoolde werknemers,... nodig → entry barriers). Switching costs: als je iets anders wilt gaan doen, kan dat veel kosten.
* Bargaining power of suppliers: de kracht die jouw leveranciers hebben op de prijs/kwaliteit van de geleverde producten. Ben jij als bedrijf in staat om van jouw leveranciers te eisen dat ze een bepaalde kwaliteit aan een bepaalde prijs leveren? Heel veel/heel weinig mogelijke leveranciers: veel leveranciers geeft meer macht want je kan makkelijker op een ander gaan (raising prices faster).

**Voorbeeld 1:** IBM was ooit een van de succesvolste bedrijven van de wereld, een heel groot Amerikaans bedrijf dat in de jaren ’70 en ’80 PC’s produceerde. Deed een heel slechte strategische keuzes: besloten om de software op hun PC’s te outsourcen (wegens dreiging van Microsoft). → Microsoft de wordt supplier van IBM. Heel weinig suppliers van besturingssystemen dus MSFT had heel veel macht. IBM kreeg concurrentie van bedrijven uit het verre Oosten (China, Japan: Lenovo, Toshiba,...). → IBM moest op prijs gaan concurreren. MSFT besliste eigenlijk over de prijs van PC’s. Klanten wouden enkel Windows dus zij beslisten over de prijszetting van PC’s op die manier. Winstmarges van IBM werden gereduceerd tot bijna

0.

**Voorbeeld 2:** Milk farmers: er zijn er enorm veel. Melkboeren leveren hun melk aan supermarkten en er zijn er zo heel veel, dus de supermarkten (zijn maar met een aantal) hebben heel veel macht op de melkboeren. Die melkboeren zorgen ervoor dat ze niet tegen elkaar kunnen uitgespeeld kunnen worden door associaties op te richten en melk in z’n geheel aan de supermarkten te verkopen om zo betere prijzen af te dingen.

* Bargaining power of buyers: als we kijken vanuit de cirkel en als we kijken vanuit het bedrijf: wat is onze bargaining power ten opzichte van onze klanten? Hoe hoog kunnen wij onze prijs zetten en hoe makkelijk kunnen wij nieuwe klanten vinden? Aantal klanten (indien maar 1 klant: veel moeilijker om veel macht te hebben in de relatie), switching costs (hoe makkelijk is het om als klant op een alternatief product over te schakelen?) → Voornamelijk gerelateerd aan productdifferentiatie? Bv iPhone: heel mainstream, maar slagen er toch in hun product zeer sterk te laten verschillen van andere telefoons. Apple verkoopt niet enkel smartphones, maar ook hun imago: productdifferentiatie, op die manier maken zij een product dat uniek lijkt voor klanten die dat imago willen kopen. De bargaining power van Apple is bijna 0: klanten willen belachelijk veel betalen.
* Threat of substitute products or services: niet bekijken als directe concurrent (smartphones zijn niet echt substituten, eerder internal rivalry). Airlines hebben rivaliteit met video conferencing bedrijven: in plaats van het vliegtuig te pakken, gewoon via Skype overleggen. Vb Amazon user profiles: zij zorgen ervoor dat substituten van hun service minder waard zijn door hun klanten zo goed te kennen dat ze extra waarde kunnen leveren: weten perfect wat de klant wilt, kunnen (bijna) voorspellen wat hun volgende aankoop gaat worden. ⇒ Het kennen van je klanten speelt een rol in de belangrijkheid van subsituutproducten.
* Rivalry among existing competitors (cirkel): als een industrie groeit, is het veel makkelijker om met concurrenten impliciet akkoorden te vinden over hoe bedrijven groeien. Bedrijven gaan relatief snel tevreden zijn. Als een markt gelijk blijft/inkrimpt & je wilt dat je bedrijf groeit, moet je omzet afnemen van concurrenten, wat vaak leidt tot prijsoorlogen die soms heel destructief kunnen zijn.

Stubborn executives: vaak gedreven door winstgevendheid/groei van ondernemingen en zo nemen ze vaak beslissingen die niet goed zijn voor het bedrijf.

**Slide 15:** Voor- en nadelen van Porter:

⊕ Generiek toepasbaar.

Weinig aanraakpunten met operationalisering van strategie: weinig richtingaangevers van wat je moet doen met jouw verhaal.

**Chapter 3**

# Les 3: 16/02/2015

## 3.1 Slides: 1b.ISstrategy

**3.1.1 Deel 3: Competitive advantage and IS: the value**

### chain perspective

**Slide 17:** Model van Porter is een nuttig strategisch model voor analyse, maar niet operationeel. Makkelijker strategisch model: Value Chain Model. Veel operationeler zicht op strategie, maar het is ook een strategisch model: model dat weergeeft waarom bedrijven bestaan en hoe deze ervoor zorgen dat ze waardevol zijn. Bedrijven geven invulling aan zo’n value chain, gaan elk van de elementen in de value chain invullen (concreet) met als doel competitive advantage/toegevoegde waarde te cre¨eren.

Belangrijk als je het model bekijkt: het bevat 5 primaire activiteiten onderaan in het oranje. Die zijn de meest cruciale activiteiten van een bedrijf. 5 zaken die een typische ondernemingen zullen doen. Lijkt heel sterk gericht op productiebedrijve, maar dat is het niet, ook servicebedrijven passen in dit plaatje. Inbound logistics zijn dan bv data. 4 gele lijnen: support activities. De 9 activiteiten in hun geheel vormen de onderneming en cre¨eren samen toegevoegde waarde (in de economie).

Zo’n waardeketen kan je losstaand analyseren, maar eigenlijk moet je ook naar het breder systeem gaan kijken van value chains, nl. value systems (1 ervan is die van jou, maar er komen heel wat value chains van anderen ingeplugd, bv die van jouw suppliers). Informatiesystemen/technologie kunnen het valuesysteem veranderen.

Bv: Dell heeft beslist om hun value systeem radicaal om te gooien [Dell’s value proposition: Dell gaat enkel en alleen orders via het internet accepteren. Klanten bestellen alleen via hun website], klantenzijde van haar value systeem te wijzigen. Klanten van competitors van de andere computermakers zijn de mediawinkels. Dell heeft gezegd van ”We hebben een superwebsite, een goed inventory systeem dus we kunnen real-time verkopen via de website zonder dat we een (externe) distributeur nodig hebben.” Een gans deel uit hun value systeem is eruit gehaald. → Desintermediation (haalde een aantal partners uit de supply chain). Door gebruik te maken van het Internet en hun productieproces zo gestroomlijnd te maken, konden ze hun value system veranderen.

**Slide 19:** Kunnen informatiesystemen een rol spelen bij de strategiebepaling en -uitvoering? Wij denken van wel: IT en informatiesystemen zijn van strategisch belang. 5 bedrijven waarbij dat effectief het geval is.

**Slide 20 ev:** Receiving process: deal met leveranciers: hoe vaak leveren ze,... Voordeel dat je kan maken: automated warehousing system: geavanceerde systemen in hun opslagplaatsen. Meer en meer op een digitale manier, geen heftruks meer. Delhaize maakt gebruik van een nieuw warehouse dat volledig geautomatiseerd is. Vroeger moesten ze voor dezelfde grootte van warehouse 100-150 mensen hebben, waar ze er nu nog 3 nodig hebben. Volledig aan de hand van computersystemen geautomatiseerd. Wat is de impact hiervan op strategie? → Niet noodzakelijk echt bedoeld, maar we kunnen nadenken wat het strategisch effect is van zo’n warehouse op de positie van Delhaize op haar concurrerende omgeving. Threat of new entrants: het kost veel om zo’n warehouse te bouwen. Delhaize slaagt er zo in om nog versere producten te voorzien. Zo kunnen ze nog kleinere pakketjes versturen naar hun winkels: kleinere pakketten → versere producten ⇒ kwaliteitssupermarkt (imago). Internal rivalry ook: je zorgt ervoor dat je kosten op langere termijn lager liggen dan die van de concurrent door met veel minder werknemers te werken. Threat of substitutes (al dan niet onbewust): versmarkten (geen directe concurrent van Delhaize), Delhaize probeert ook het risico dat klanten naar de versmarkten gaan te counteren door te zorgen voor dagverse producten. Restaurants zijn ook een indirect substituut: als wij veel vaker op restaurant gaan, gaat de omzet van Delhaize naar beneden. Delhaize laat dus toe om sterk te spelen op de versheid en dat aan klanten aan te geven en daarom moet je niet per s´e op restaurant gaan als je vers wilt en ook niet naar versmarkt.

**Slide 22 ev:** Operations: alles wat te maken heeft met het produceren (goederen *en* diensten). Alles wat te maken heeft met computergestuurde productie, meer en meer gaat men vertrouwen op computersturing om ervoor te zorgen zodat je met minder werknemers een bepaald product kan produceren. Ook alles wat te maken heeft met automatisering (operational efficiency). Ryanair: strategie: lage kosten. Ryanair is een bedrijf dat opereert aan de hand van een lage-kostenstructuur. Hoe slagen zij daarin? → Personeel wordt minder betaald, veel flexibeler werken (ja?), gebruiken IT op zo’n manier dat een lagekostenstructuur werkt. Bv voor hun website slagen ze erin dat ze zo weinig mogelijk extra kosten maken om hun klanten te bedienen: geen instapkaarten (zelf afdrukken), tenzij je die niet bijhebt en dan betaal je ervoor bij. Website wordt ook gebruikt (buiten operational perspectief) in marketing en sales: hun marketingcampagnes zijn heel sterk gelinked aan hun website, daar brengen ze klanten samen. Prijszetting kan heel dynamisch geconfigureerd worden. De manier waarop Ryanair met die website werkt en die configureren die vorm van prijstransparantie heel laag is (is al beter door wetgevingen). Keken ook naar welk soort PC je gebruikt om prijszetting te bepalen. Internal rivalry, entrance barriers, buying power van klanten (is heel laag), threat of substitute products.

**Slide 24 ev:** Outbound logistics: alles wat te maken heeft met web-gebaseerde verkopen: op welke manier je je klanten bedient, voyage estimated (???) kan je hier ook vermelden als hoe IT mee een competitief voordeel kan leveren. Amazon: op het vlak van outbound logistics in Amazon waarschijnlijk het bedrijf waar je het meest van kan leren. Op operationeel niveau niet zo verschillend van andere bedrijven, maar dus wel op vlak van R&D, outbound logistics. Gaan nieuwe vormen van bedrijfsvoering introduceren (drones). Amazon claimt dat zij producten naar jou kunnen verzenden nog voor jij betaald hebt. Het bedrijf is in staat om gebruik te maken van info van ons doordat wij daar kopen en op den duur weet Amazon zoveel over ons dat ze in staat zijn om producten naar mensen te versturen nog voor ze gekocht zijn. Amazon is dus heel sterk in outbound logistics. Is van strategisch belang voor Amazon: threat of new entrants (ze slagen erin kwaliteitsgaranties te leveren aan klanten die andere bedrijven gewoonweg niet kunnen bieden).

**Slide 26 ev:** Alles wat te maken heeft met promoties, market resales. Targeted marketing: heel veel bedrijven zetten hierop in. Bv Colruyt: speelt op lage kosten (alle zaken die ze doen zo effici¨ent mogelijk en aan lage kosten) en zorgt ervoor dat dat het aanspreekt bij klanten. Investeren daarom heel sterk in marketing & sales: 1,6 miljoen gepersonaliseerde reclamefolders met coupons en discounts die die klant koopt of voor producten die de klant misschien eens zou willen proberen. Ze hebben heel sterk gespeeld op het investeren van infostructuur die ervoor zorgt dat ze info kunnen opslaan over klanten en deze te gebruiken: een heel specifiek voordeel, waarde voor de klant. Op die manier spelen ze op het strategische niveau, zorgen ervoor dat threat of new entrants laag is.

**Slide 28 ev:** Customer service: alles wat te maken heeft met after-sales: helpdesks, garanties,... Op welke manier kan IT daar competitief voordeel bieden? → CRM.

Bv: Booking.com: soort van concurrent van reisorganisatoren, maar zuiver digitaal, enkel bruikbaar via het Internet maar toch directe concurrent van reisbureau. Op vlak van CRM: Booking heeft heel veel info over zijn klanten dat toelaat om een heel specifieke manier van hun website aan de klant te tonen: tonen enkel hotels zodat enkel degenen die het meest aantrekkelijk zijn voor de klant getoond worden. Levert waarde aan klant doordat zijn keuze vergemakkelijk wordt. Houdt betalingsinfo bij, alsook opinies over hotels. Hierdoor gaan klanten telkens willen terugkeren naar Booking.

**Slide 30 ev:** Je kan via human resources zorgen dat je voorstaat op andere bedrijven door competent personeel aan te nemen. Via IT voordeel behalen: employee benefits intranet: veel werknemers hebben een persoonlijke voorkeur over hoe hun loon wordt uitbetaald: werknemers kunnen hun eigen beloningspakket samenstellen (wel die verzekering, geen auto, wel gsm,...). HR analytics: gaat erover dat je kan voorspellen welke skills, talenten jouw organisatie nodig gaat hebben, dat op een optimale manier te voorspellen, hoe je die skills moet kunnen inzetten. Kunnen we info over bestaande werknemers inzetten om te weten welk soort werknemers we gaan nodig hebben?

R&D: niet noodzakelijk altijd heel strategisch, maar CAD-systems: nieuwe manieren om aan productontwikkeling te doen en zo voordeel halen uit (snellere) lancering van producten.

Procurement: heeft te maken met het core product met wat je wilt produceren (niet zeker hiervan!). Ook daar slagen bedrijven erin om optimalere vormen van procurement te organiseren met behulp van IT.

Reverse auctions: jij gaat als klant aangeven wat je wilt en aanbieders gaan aan jou voorstellen doen: 1 klant contacteert meerdere suppliers om een procurement te doen voor een materiaal dat nodig is.

**3.1.2 Deel 4: Linking business strategy and IS strategy:**

### Henderson & Venkatraman’s strategic alignment model

**Slide 35:** Model van H&V (slide 36): componenten in het rood: bedrijfsstrategie, ICT-strategy, operational infrastructure & processes, ICT infrastructuur & processen. Je moet specifieker gaan kijken naar de afstemming van de verschillende componenten. Kijk of de bedrijfsstrategie (Hoe positioneert een bedrijf zich ten opzichte van zijn/haar concurrenten? → In de markt ⇒ extern) afstemt op het operationeel niveau (intern). → De fit bestaat erin om de interne operationering af te stemmen op de strategische fit. H&V hebben enkel ICT strategy toegevoegd.

Traditioneel: de bedrijfsstrategie en de operationele strategie bepaal je en die zorgen er samen voor dat ICT infrastructuur mogelijk wordt. H&V hebben gezegd dat die 3 (zonder ICT-strategie) enkel onvoldoende zijn, je moet ICTStrategy toevoegen: je gaat kijken op het vlak van de ICT markt, de evoluties op het vlak van IT. ICT-strategie is er om keuzes te maken met betrekking tot de ICT Market Place. → Mogelijke keuzes op vlak van IT ten opzichte van de concurrenten.

Je moet ervoor zorgen dat externe en interne ICT zaken op elkaar afgesteld zijn. Functionele fit: ervoor zorgen dat je ICT-keuzes afgestemd zijn op de operationele infrastructuur. Ook de afstemming van bedrijfsstrategie op ICTstrategie.

**Slide 37:** Hoe realiseer je extern en intern uw strategie¨en?

**Slide 38:** Hoe positioneert ons bedrijf zich ten opzichte van de ICT market place en hoe passen we daarop een strategie toe? Die strategie is extern maar wordt gerealiseerd door de keuzes intern door te trekken, te operationaliseren. → IT niet puur zien als ondersteuning. Je kan op basis van IT strategie ook doen aan business strategie.

**Slide 38:** Functional fit == strategic integration.

**Slide 39:** Problemen met functional fit: management: typisch verantwoordelijk voor bedrijfsstrategie. Als we naar de fit gaan kijken tussen bedrijfs& ICT-strategy, zijn die vaak niet op elkaar afgestemd, dus krijg je problemen. Topmanagers die nog steeds zeggen dat ICT aan de CIO moet overgelaten worden en aan de IT mensen.

Increasing awareness: werknemers vandaag de dag zijn meer en meer bewust van wat IT kan, dan is het zo dat je dat in een bedrijfscontext ook wil. Werknemers in bedrijven zijn zich meer en meer bewust van de mogelijkheden in IT en willen dat vertaald zien in hun dagdagelijkse bezigheid. Dat is niet altijd mogelijk voor organisaties om dat te ondersteunen.

Impatient to use IT: heel vaak, als er gekozen wordt om digitaal te innoveren, zijn eindgebruikers heel gedreven om het te gebruiken. Op die manier zetten ze de ontwikkeling onder druk waardoor er weinig tijd is om de systemen te gaan testen. → Groot verschil in hoe men het wil gebruiken en hoe het gecodeerd is.

**Slide 40 ev:** Hoe ga je hier als bedrijf mee aan de slag? Hoe ga je keuzes maken? H&V beschrijven 4 mogelijke paden voor business & IT alignment ⇒ 4 paden doorheen het model die bedrijven kunnen volgen om alignment te realiseren:

1. Strategic execution: je vertrekt van jouw business strategy die de operationele omgeving bepaalt en zo worden al dan niet achter de schermen keuzes gemaakt over ICT infrastructure: verticale & horizonale fit → afstemming tussen IT en business.

Bv De Lijn: veel investeringen in IT, hun strategie kan je beschrijven als strategic execution: bedrijfsstrategie is nog steeds dezelfde, hebben operationele omgeving (bussen, ticketverkoop,...), maar maken gebruik van IT om de operationele omgeving te ondersteunen (website, ticket via sms,...). Om dat nog beter te doen, implementeren ze nog betere systemen (websystemen) om de ticketverkoop nog beter te laten verlopen.

1. Technology transformation slaat erop dat je kijkt naar de bedrijfsstrategie en dan gaat kijken hoe je je IT-strategie daar best op kan afgestemd kan worden voor je gaat kijken naar de operationele omgeving (via ICTinfrastructure).

Bv: Colruyt: retailer met lage-kostenstructuur, zetten daar heel hard op in (bedrijfsstrategie), nu kijken zij ook naar hun ICT-strategie, hoe zij die lage-kostenformule zo goed mogelijk kunnen ondersteunen met ICT. Hoe? → Producten/diensten aankopen op vlak van ICT-strategie. ICT-market place voldeed niet aan wat zij verwachtten (15 jaar geleden, te duur etc) → hebben een bedrijf gecre¨eerd: Dolmen (nu Real Dolmen) om hun strategie te cre¨eren: support van hun lage kostensysteem. Hebben dus hun eigen intern (nu niet meer) IT-bedrijf opgericht om zelf een invulling van de (externe) IT te cre¨eren.

1. Competitive potential: vertrek vanuit de strategische positionering van jouw IT ten opzichte van de mogelijkheden die er zijn: zelf eerst op vlak van ICT competitieve voordelen inschatten, die capaciteiten ga je gebruiken om een bedrijfsstrategie te bepalen en die bepaalt uw operationalisering.

Bv: Amazon: zeer sterke focus op IT (distributie, verkoop, CRM,...).

1. Service level: vertrekt van ICT-strategie: eerst positionering extern op het vlak van ICT, dan ga je zelf eerst een operationalisering daarvan realiseren, intern. Vervolgens ga je merken dat die operationalisering waardevol is (service level). ⇒ Vertrek van externe positionering, operationaliseer de voordelen en dan merk je dat dat systeem waardevol is voor andere bedrijven. Op operationeel niveau is het systeem dat jij gebouwd hebt waardevol voor andere bedrijven. Die infrastructuur kan je gaan verkopen aan andere bedrijven.

Bv: SAP: ontwikkelt software puur en alleen voor bedrijven om de operationalisering voor bedrijven te gaan realiseren.

**Slide 49:** Conclusie: de 4 paden doorheen het model zijn heel belangrijk. Merk op dat het niet meer het meest actuele model is. Het geldt nog steeds in veel gevallen, maar er zijn ondertussen andere strategische modellen gebouwd die de link tussen business en IT beter beschrijven. Tekst van Bharadwaj is niet te kennen, maar toont dat het anders kan bekeken worden. → We *mergen* de verschillende strategie¨en. In de wereld waarin we nu leven moeten we die zaken *mergen* → digital business strategy.

(1C : slides? Hebben we dat gezien?)

**Chapter 4**

# Les 4: 20/02/2015

## 4.1 Slides: 2A. Creating Value with IT

Hoe gaan we informatiesystemen bouwen? Begin met requirements: lastenboek (Wat moet erin zitten aan functionaliteit, maar ook niet-functionele zaken?). Er zijn 3 grote bronnen van requirements: bedrijf, klant en gebruiker.

Bv Toledo: bedrijf maakt software Blackboard en zij hebben een aantal vereisten ten aanzien van de software, deze vereisten gaan heel sterk gelinkt zijn aan de vereisten die de klant heeft. De klant in dit geval is KULeuven want zij kopen bij Blackboard het product, maar de gebruikers, dat zijn wij. En wij hebben allemaal verschillende vereisten (bv makkelijk in gebruik, veel verschillende features), maar de KUL zal bv meer belang hechten aan de kostprijs. Vaak zorgen bedrijven ervoor dat er een lock-in plaatsvindt, zodanig dat gebruikers die er al in ge¨ınvesteerd hebben niet zomaar (kunnen) gaan switchen.

Het proces van ontwikkeling verloopt via bepaalde stappen (analyse van vereisten, technisch ontwerpen, implementeren, testen, verbeteren).

Eens een systeem in gebruik: maintenance: verbeteren en uitbreiden.

**Slide 4:** Strategie¨en zijn manieren om tot uw requirements te komen. Value chain: verschillende processen die elkaar opvolgen waaruit ook vereisten voortvloeien. Heel belangrijk dat bij het uittekenen van een strategie je dat doet in lijn met business strategy.

**Slide 5:** Elk project gaat leiden tot een subset van requirements die wordt gerealiseerd via analyse, uitwerking,... → Hoe realiseer je waarde over alle projecten heen, maar ook binnen elk project? Hoe zorg je dat elk project waardevol is voor het bedrijf?

### 4.1.1 Business Value of IT

**Slide 7:** Als men kijkt naar de waarde van IT, dan kijkt men naar verschillende zaken die waarde kunnen genereren. Als we spreken over ICT (zeer ruim containerbegrip) kun je je afvragen wat ICT is. Waar veel mensen aan denken is de technologie, maar je hebt ook nog de mensen (technische kennis en begrijpen van de business), je moet problemen van een bedrijf kunnen oplossen dus mensen moeten niet alleen technologische problemen oplossen, maar ook begrijpen hoe het bedrijf beter kan functioneren en ze moeten ook probleemoplossend denken.

Relationship asset: business IT aligment is heel belangrijk: als je een goede relatie hebt tussen business en IT, wat op zich zeer waardevol is.

**Slide 8:** Verschillende elementen inzetten voor verschillende doelen. Kijk naar de dimensies van de waarde die je kan cre¨eren:

* Operationele waarde: weg van papier, maar via IT → operational excellent: bedrijfsprocessen sneller en vlotter laten verlopen, met minder fouten en aan een lagere kost → waarde op de werkvloer in de concrete, dagelijkse waarde van het bedrijf.
* Tactical: betere dienstverlening naar klanten toe. Relatie met leveranciers en partners verbeteren. Het kan zijn dat een bepaald proces niet per s´e goedkoper wordt via IT! Maar je customer service gaat erop vooruit want klanten appreci¨eren de levering vs het zelf afhalen.
* Strategisch: competitief voordeel dat bestendig is.

⇒ Dimensies waarop je waarde probeert te cre¨eren. Als men IT value wil gaan meten wordt het moeilijk om de strategic & tactical value van iets te meten.

**Slide 9:** Uitspraak van Solow: kritiek erop na onderzoeken (sommigen zeiden wel waar, anderen niet), doorheen de jaren komt men tot het besluit dat de paradox niet correct is. Latere studies hebben aangetoond dat er een positief verband bestaat tussen investeringen in IT, productiviteit & business value. Verklaringen:

* Moeilijk om te meten wat de voordelen zijn van IT. Vroeger moest je keuzevakken aangeven via papier. Met de invoering van het ISP: meer werk voor studenten (meer tijd besteden om alles in te geven), het nakijken van het papier ging ook sneller dan op PC, dus zeker in het begin (bij de invoering van het ISP) zou het lijken dat papier sneller en effici¨enter was dan nieuwe ISP-systeem → negatieve evaluatie. Maar doordat alles van in het begin van het academiejaar kan gebeuren, zijn er mensen die er net voordeel bij hebben: professoren weten sneller wie er ingeschreven is. Die toegevoegde waarde is moeilijk te meten.
* Time lags: eerste jaar dat het systeem wordt ingevoerd: nachtmerrie (vol fouten, onhandigheden,...). Eerste jaar/jaren geen benefits, meer investering dan voordeel in het begin.
* Redistribution: studentensecretariaat heeft meer werk, proffen, decaan,... minder ⇒ som = 0.
* Mismanagement.

De investering is vrij goed te meten, maar de waarde is moeilijk te meten. Als je het op de juiste manier probeert te meten, kan men een positieve link vaststellen.

**Slide 10:** Geen eenduidigheid, geen intangible benefits, geen rekening houden met bepaalde risico’s.

Management: value creation models (hoe cre¨eer je waarde?), performance measurement models.

**Slide 11:** Cyclus van define, measure, analyze, improve, control.

**Slide 12:** Internal process efficiency.

**Slide 13:** IT balanced score card: voor bepaalde projecten kijken hoe ze bijdragen tot de organisatie.

Het grote probleem hiermee: het bepalen van de KPI’s: hoe future orientation meten? → Zeer moeilijk!

Balanced score card geeft een instrument maar het blijft instrument met zeer veel mankementen en moeilijkheden in de toepassing ervan.

**Slide 15:** Modellen gezien: proberen via management praktijken aandacht te trekken op gebruiken die waarde kunnen cre¨eren,... → Combinaties van SS, BSC, COBIT: hopen dat het redelijk goed zal gaan.

**4.1.2 If we build it, will they come?**

**Slide 18:** Technology acceptance: gaan kijken wanneer een systeem gebruikt wordt: heeft niet rechtstreeks te maken met het voordeel dat een systeem biedt. Wat maakt dat bepaalde mensen iets wel/niet willen gebruiken?

**Chapter 5**

# Les 5: 23/02/2015

## 5.1 Slides: 2A. Creating Value with IT

**5.1.1 If we build it, will they come?**

**Slide 17 & 18:** Toledo en Facebook: beiden informatietechnologie. We gaan nu naar technology acceptance kijken op gebied van de gebruiker. We zien dat Facebook meer aanvaard is dan Toledo.

**Slide 19:** Wat drijft mensen tot een bepaald gedrag? Uw keuze om iets te doen (studeren, spieken, naar een fuif gaan,...) is allemaal gebaseerd op uw verwachting van het resultaat en de waarde die je hecht aan het resultaat. Bv: in welke mate ben je bereid om een avond op te geven aan studeren in favor of fuif? Is ook een kansberekening: als ik naar de fuif ga, wat is dan de kans dat ik geslaagd ben? Combinatie van kans en waarde maakt dat je kiest om al dan niet thuis te blijven om te studeren.

**Slide 20:** Theory of reasoned action: bewuste keuzes die je maakt: uw intenties om iets te doen worden bepaald door je attitude ten aanzien van het gedrag (combinatie van wat je gelooft over bepaalde gevolgen en de evaluatie van die gevolgen - wat hierboven werd besproken) en subjective norm (normative beliefs: wat je denkt in termen van normen, wat past en niet past en uw motivatie om daaraan te voldoen.

Bv: met kapotte jeans rondlopen etc. behoort niet tot de normen en ik wil voldoen aan de norm op vlak van dresscode, dus ga ik geen kapotte jeans aandoen - je gaat hierbij ook uit van de verwachtingen van de omgeving.

**Slide 21:** Voortbouwend op die theorie: subjectieve norm weglaten (er is een directe invloed op de behavioral intention, maar ook een indirect effect via attitude, subjectieve norm zit al in values en beliefs in attitude), maar vroeg zich dan wel af wat de attitude bepaalt. Voor technologie: perceived usefulness and perceived ease of use → correleren met uw attitude naar het gebruik van de technologie en bepalen uw intentie om iets te gebruiken. Gebruiksgemak bepaalt niet alleen uw attitude of behavioral intention, maar ook de wil om het te gebruiken.

**Slide 22:** Key variables:

* Perceived usefulness: wordt bekeken vanuit het perspectief van de gebruiker, niet van het bedrijf. Als gebruiker meet je het nut voor jou persoonlijk. Meet de mate waarin een persoon denkt dat het gebruik van de toepassing hem/haar gaat helpen bij het uitvoeren van een taak. Heeft invloed op attitude en behavioral intention.
* Perceived ease of use: gebruiksgemak: bijkomend ondersteund door cost/benefitparadigm: kost heeft onder andere te maken met ease of use, met self-efficacy theory (mate waarin jij vlot kan omgaan met technologie).

**Slide 23:** Beperkingen aan het onderzoek:

* Wanneer er wordt gemeten naar behavioral intention: is gebeurd aan de hand van een onderzoek waar de gebruikers zelf moesten aangeven of ze iets gebruikten of niet (het gebruik werd dus niet werkelijk gemeten, werd gewoon gevraagd → gebruikers antwoorden niet altijd overeenkomstig de realiteit).
* Onderzoekers hebben de neiging om studenten te gebruiken als steekproef (veel makkelijker), maar hoe representatief zijn studenten voor de hele bevolkingsgroep?
* Gaat steeds over onderzoek waarbij vrijwillig gebruik wordt gemaakt van bepaalde zaken. Het model is dus niet van toepassing op onvrijwililg gebruik.

⇒ Unified theory of acceptance is daaruit gekomen als antwoord.

**Slide 24:** Neemt variabelen van Davis met enkele extra variabelen:

* Perceived Usefulness: hoe hard gaat dit mij helpen?
* Effort expectancy: in welke mate ga ik een inspanning moeten doen om dit medium te gebruiken?
* Social influence: er is een invloed van de mensen rondom jou, dus maak onderscheid tussen groepsdruk en de eigen keuze.
* Facilitating conditions: elementen die het voor u makkelijker maken om iets te gebruiken (bv aanwezigheid helpdesk, voorzien opleiding, handleiding,...). → De mate waarin jij denkt dat wanneer je in de problemen komt, je ergens terecht gaat kunnen voor hulp bepaalt mee of je iets wilt gebruiken.

**Slide 26:** De eerste 3 (PE, EE, SI) bepalen de behavioral intention. De mate waarin je naar gebruik overgaat heeft ook te maken met facilitating conditions. De vier vakjes onderaan zijn moderatoren: bepalen in welke mate die verschillende invloed hebben op de intenties. → Slide 26 is daar een vb: lees die tabel van links naar rechts.

### 5.1.2 Information System Success

**Slide 27:** 3e model van DeLone & McLean: hebben verschillende dingen gemeten (**slide 28**) en zijn gaan kijken wat het succes bepaalt en wat niet.

**Slide 29:** Systeemkwaliteit: allemaal variabelen die ergens iets te maken hebben met gebruiksgemak (slide 29: manieren waarop je systeemkwaliteit kan meten. Zeggen niet *hoe* je dat moet meten, dat moet je zelf bepalen).

**Slide 30:** Informatiekwaliteit: als je ergens naartoe gaat en je wenst info te vinden, ga je dat niet alleen beoordelen op het gemakkelijk terugvinden, maar ook of je voldoende terugvindt, zoals verwacht. Kijken of het relevante info is, of het aan je verwachtingen voldoet. Je kijkt ook naar de inhoud. → Key antecedent of user satisfaction: user satisfaction wordt heel hard bepaald door de inhoud die aangeboden wordt. Heel veel dimensies van info die je kan meten: vertrouwen, accuraatheid,...

**Slide 31:** Service quality: facilitating conditions: mate waarin je gelooft dat je geholpen gaat worden/dienstverlening krijgt.

SERVQUAL: manier om service quality te meten.

**Slide 32:** Intention to use: mate waarin je denkt een systeem te gaan gebruiken. Vaak een surrogaat voor het gebruik. Als je het gebruik niet kan meten (systeem is bv nog niet operationeel), dan kan je de intention to use gebruiken als surrogaat hiervoor. Duidelijk onderscheid tussen vrijwillig & verplicht gebruik. Bij verplicht gebruik kan je het niet gebruiken (moet je dit laten vallen). Wanneer mensen de keuze hebben om het niet te gebruiken, kan je het wel meten.

**Slide 33:** User satisfaction: meten in welke mate de gebruiker tevreden is over iets. Ook wanneer de gebruiker verplicht is kan je kijken of die tevreden is of niet. Probleem bij het meten van gebruikerstevredenheid is dat het overlappend kan zijn met informatie- & systeemkwaliteit.

**Slide 34:** Net Benefits: voordeel dat uit iets wordt gehaald. Volgens DL&McL bevat dit zowel het voordeel voor de organisatie als voor het individu. Je moet dus kijken naar het totale voordeel. Kan een voordeel voor maatschappij bieden, of voor externen aan het systeem. Je kan heel veel dingen vatten onder die noemer van net benefit. Ze zeggen ook hier niet hoe je net benefit moet meten, wel dat het gemeten moet worden. Je kiest de manier die voor jouw studie het meest relevant is, met alle moeilijkheden die daarbij komen kijken.

**Slide 35:** Relatie tussen verschillende elementen: groene elementen gaan intention to use & user satisfaction be¨ınvloeden. Intention to use gaat use bepalen. Wisselwerking tussen intention to use & user satisfaction. Intention to use/use & user satisfaction bepalen de net benefits. Deze bepalen op hun beurt ook de ITU en US. ⇒ Veel relaties tussen de verschillende elementen. Als iets nuttig is (NB is hoog) ga je het willen gebruiken en ga je waarschijnlijk ook tevreden zijn. Tevredenheid wordt mee bepaald door de groene dingen (onrechtstreeks bepalen de groene dingen dus of je tevreden bent en of je van plan bent om iets te gebruiken).

Tekortkomingen:

* Use kan ook invloed hebben op (on)volledigheid van info. Bv een systeem dat vrijwillig in gebruik is en niemand voegt iets toe, dan zal de informatiekwaliteit zeer laag zijn. Intentie om er dan naar te kijken zal zeer laag zijn (zie ook use case 1). Het gegeven model capteert dat niet.
* Staat niet duidelijk hoe je elke parameter moet meten.

Er is een vervolgonderzoek geweest waarin ze keken naar hoe je alles moest meten etc: is nog te vroeg om daar een duidelijke trend in te zien en om er besluiten uit te kunnen trekken. Wat je wel kan onthouden is dat de 3 groene blokken heel belangrijk zijn in het bereiken van informatiesysteemsucces. Net benefit: bewaken van het voordeel van in het begin is ook heel belangrijk. Zelfs al zijn de relaties niet altijd even duidelijk, je mag ervan uitgaan dat groen & oranje aan de basis liggen van informatiesysteemsucces.

### 5.1.3 Case Study

**Road Sign Database**

**Slide 37 ev:** Case van verkeersbordendatabank: vrachtwagen reed onder een te lage brug omdat er geen verkeersbord was dat aangaf wat de vrije hoogte was. Heeft heel Vlaanderen plat gelegd wegens kettingreactie (heel de autostrade lag stil).

In 2008 is er een project gestart. Basisidee: databank maken met alle verkeersborden in heel Vlaanderen zodanig dat ten alle tijde gecontroleerd kan worden of de nodige borden al dan niet aanwezig zijn (bv bij het bouwen nieuwe brug). Initi¨ele kostschatting van 20 miljoen, waar men heel ver is overgegaan. Men is er nog steeds aan aan het werken om het systeem op poten te krijgen. Er is al een veelvoud van het geld ingestoken. Een onderzoekster heeft in 2014 telefonische enquˆetes & interviews gedaan (Wie gebruikt het systeem? Waarom wel/niet?). De resultaten staan in de kader slide 37. 66: ooit gebruikt, maar mee gestopt. De bedoeling is: je hebt een databank op Vlaams niveau waarin gemeentes inloggen en verkeersborden invullen/weghalen.

**Slide 38 ev:** Als je de elementen overloopt aan de hand van UTAUT & DL&McL, wat zijn dan de oorzaken van het falen van het project?

* Performance expectancy: voor een stuk was initi¨eel de hoop dat dit het werk zou vergemakkelijken. In het begin was er een zeker enthousiasme, maar eens het systeem gemaakt was, volgde de desillusie: geen toegevoegde waarde voor het werk binnen de gemeente. Bovendien kwam er na verloop van tijd een goedkoop & makkelijk alternatief: Google Street View. Tegen de tijd dat het project van de verkeersborden ontwikkeld was, was GSV er. Gemeenten keken zo of een bord er stond of niet. Initi¨eel werden er beloften gedaan over bijkomende toepassingen in verband met wetgeving, maar uiteindelijk is dat dus niet gerealiseerd. Features die potenti¨ele toegevoegde waarde hadden zijn niet gerealiseerd → performace expectancy naar beneden.item
* Effort expectancy: er werden opleidingen gegeven om het systeem te gebruiken, maar die werden geannuleerd; mensen kwamen naar de opleiding, maar lesgever kon niet inloggen en dat verschillende keren. Het systeem was zo slecht gemaakt dat zelfs de lesgever niet ingelogd raakte. Ook achteraf, na een eventuele opleiding, kwamen er terug problemen bovendrijven. Bovendien crashte het systeem regelmatig waardoor al het werk kwijt was. Hoe meer gebruikers op het systeem inlogden, hoe trager het systeem werd. → Verwachting kwam op dat het systeem zeer moeilijk te gebruiken was en te veel technische problemen had. Effort expectancy werd dus zeer slecht beoordeeld. ⇒ Intention to use werd zeer laag.
* Social influence: de geruchtenmolen dat het zo slecht gemaakt was kwam op gang. → Bijkomend negatief effect.
* Facilitating conditions: er was een helpdesk, maar door vele problemen was die overbelast. Vragen konden niet beantwoord worden omdat er veel te veel waren. Facilitating conditions waren dus ook negatief. ⇒ Negatief op alle dimensies die er zijn.
* Moderating factors: bleek weinig invloed te hebben.
* Voluntary use: gemeentes werden nooit verplicht om het te gebruiken, anders was het misschien anders gelopen.

**Conclusie (slide 40):** Zowat alles slaagt tegen.

**Slide 41 ev:** Vanuit DL&McL:

* System quality: slecht: performantieproblemen, het systeem is niet betrouwbaar, niet flexibel in gebruik, traag, zeer moeilijk in gebruik,...→ In het algemeen heel negatief beoordeeld.
* Informatiekwaliteit: heeft tot bedoeling om data te verzamelen. Maar de meerderheid van gemeentes gebruiken het niet, dus er zit geen info in. → De informatiekwaliteit is laag omdat het systeem onvolledig is.
* Service kwaliteit: helpdesk is competent en vriendelijk, maar overbelast dus we krijgen niet ge gewilde/gevraagde service.
* Intentie tot gebruik: bepaald door de vorige factoren. Sommige mensen wouden het echt wel gebruiken, maar hoorden dat het slecht was (bad reputation, is hier nochtans niet inbegrepen! → Tekortkoming van DL&McL). Er was geen verplichting om het te gebruiken (voluntariness of use zit hier ook niet in, ook een tekortkoming!). Moest het verplicht geweest zijn/had men er geld voor gegeven, was het waarschijnlijk meer gebruikt geweest. Er zijn ook alternatieven die beter zijn, maakt dat ze dit minder willen gebruiken.
* User satisfaction: initi¨eel vonden mensen het een goed idee, maar uiteindelijk is het niet bruikbaar door alle andere factoren. Was dus initi¨eel heel hoog, maar na ingebruikname heel hard gezakt.
* Net benefits: GSV was er in Europa nog niet in 2008, maar in 2011 begon men de Belgische steden toe te voegen. Had men in 2008 kunnen voorzien dat GSV in Europa zou komen? Sommige Google projecten blijven niet leven. Een vraag die je je bij GSV kan stellen is hoe up-to-date dat is. Een databank heeft dus wel voordelen. Maar heel kleine gemeentes weten waar de borden staan (hun argument). Grotere steden hadden al eigen databank en vonden een Vlaamse databank dus niet nodig. Wouden die wel doorgeven aan de grote databank, maar dan wel in hun eigen formaat. Automatische overdracht bestond niet en was niet voorzien. Op het Vlaams niveau is het misschien nuttig om een overzicht te hebben van verkeersborden, maar veel gemeenten kijken enkel binnen de grenzen van hun eigen gemeente/stad/provincie.

Net benefits: bij de start van het project had men beter moeten nadenken over waar het voordeel juist zit, wie er voordeel bij heeft en wie de kosten/effort van de invoer draagt. Als je als gemeente de effort moet nemen van alles in te voeren, maar er geen nut bij hebt, gaat uw motivatie zeer laag zijn. ⇒ Zorg voor een nut voor individuele gemeentes, niet alleen voor de Vlaamse overheid.

**Slide 47:** In plaats van dat information quality alleen op ITU invloed heeft, ook op information quality zelf (zie rode pijlen) ⇒ implosie.

**FEB student help desk**

**Slide 49 ev:** Hervormingen binnen de faculteit. Veranderingen over welke vakken je kan/mag volgen. Leerkrediet ingevoerd. Vragen daarover → helpdesk nodig. Lange wachtrijen aan het secretariaat, mailbox overvol,... tweede alternatief: online helpdesk.

* Alternatief 1: e-mail:
  + Studentperspectief: ease of use & usefulness zijn heel hoog → positieve attitude, hebben dus de intentie om dat te doen en doen dat ook heel vaak. Door de hervormingen waren er heel veel vragen en doordat alles positief is, waren er enorm veel emails.
  + Perspectief staff members: gebruiksgemak is goed en utility is ook goed → nuttig & goed systeem. Positieve attitude tot gebruik & willen het gebruiken, maar doordat er zoveel mails kwamen, werd het gebruiksgemak getemperd → uitpuilende mailbox & gigantisch veel overuren. ⇒ De oorspronkelijke attitude ten aanzien van mail werd getemperd. De technologie is handig en goed te gebruiken, maar er kwamen te veel mails binnen.
  + Management perspectief: zij gebruiken die mails niet, forwarden die naar een staff mederwerker, antwoorden meestal zelf niet. Voor hen speelt die ease of use/utility minder/niet mee. Zij zijn vooral ge¨ınteresseerd in informatie. Ze vragen zich af hoeveel vragen er binnenkomen en hoeveel tijd er nodig is om de vragen te beantwoorden. Studenten protesteren dat ze te lang moeten wachten op antwoord (zijn ze te veeleisend of is het echt zo?). Onmogelijk om hier info over

te krijgen, want ze hebben geen plaatje over wat er aan de hand is → management probleem want ze kunnen moeilijk ingrijpen. DL&McL is hier relevanter.

* Online helpdesk kwam er: studenten: geen acceptatie van de technologie: student vindt het invullen van een webformulier nutteloos & ongemakkelijk. E-mail is veel makkelijker. Utility van formulier is laag. Alles dus negatief. Staff members vonden dat veel te onhandig in het begin. Hebben begrip voor utility: overzicht, maar laag gebruiksgemak dus toch negatieve attitude ten opzichte van de helpdesk. Management: ease of use: zij moeten het niet gebruiken, maar alle info over de vragen staat in de databank & we weten nu hoeveel vragen er zijn. We weten hoeveel tijd er verloopt tussen het binnenkomen van de vraag en het oplossen → gemiddelde oplostijd van een vraag is bekend. We weten hoelang een bepaalde medewerker doet over het beantwoorden van vragen. → Perceived usefulness is zeer hoog! Het globale plaatje was er niet via e-mail (als iemand een nee kreeg voor een vrijstelling bij staff, mailde die gewoon naar de decaan en die zei eventueel ja → wie is er dan verantwoordelijk voor?).

Staff bleef antwoorden op mails in plaats van studenten te verplichten naar helpdesk te gaan. Seminarie geweest → voorstellen om het systeem te verbeteren:

* Er moest iets gedaan worden aan de ease of use voor staff members (moesten zelf invullen waar de vraag over ging etc). → Student moest zelf invullen waar de vraag over ging.
* Voluntariness of use: verplicht gemaakt om de helpdesk te gebruiken. Geen e-mail meer! Voor studenten ook verplicht gemaakt: geen antwoord meer op mails.

⇒ Perceived utility voor medewerkers verhoogd. Ook als een collega ziek was, kon een student geholpen worden door de databank. Globale plaatje bekeken & analyse gedaan over welke vragen veel gevraagd werden, dit probeerde men dan duidelijker te maken via een infosessie of een manual. Studenten konden minder shoppen → voordeel voor staff! ⇒ Bereidheid tot gebruik voor staff is hoger.

In het begin: menselijke dispatcher: voor wie is deze vraag? In het nieuwe systeem: student moet de vraag zelf invullen en al zaken aangeven (waarover het gaat) → algoritme dat op basis van kenmerken van die vraag beslist wie die vraag moet beantwoorden, dus veel sneller dan een persoon die dat moest verdelen ⇒ performantieverhogend (productiviteit stijgt, antwoordtijd daalt). → Utility voor student zou ook omhoog moeten gaan want de antwoordtijd gaat naar beneden. Voor staffmedewerkers meer voordeel dan voor student (student moet een heel formulier invullen in plaats van gewoon een mail sturen). Door te werken op die verschillende manieren, kan je de intentie tot gebruik verhogen.

**Slide 58:** Kijken wat een model bruikbaar maakt is nog steeds in ontwikkeling (niet zeker of ze dat zei). Wat je kunt onthouden is dat het weigeren om een systeem te gebruiken de benefit van het systeem naar beneden haalt. Je kan een heel mooi systeem maken, maar als niemand het gebruikt, heeft het geen nut. ⇒ User acceptance is belangrijk!

Open vraag: design perspectief: wat maakt een systeem gebruiksvriendelijk/dat een systeem wordt aanvaard door gebruikers? → Gemak om het te gebruiken (onderzoek wordt daarnaar gedaan!) & perceived utility. Die 2 elementen in gedachten houden, voor individuele gebruiker & organisatie → gebruik zal wel komen.

Overgang van oud naar nieuw systeem: change management: wat je kent, vind je altijd makkelijker. Iets nieuws wordt zowat altijd gezien als moeilijk. Begeleiden van veranderingen! → Zien dat er voldoende training is bij het gebruik en voldoende toelichten waarom het systeem nuttig is (time lag!), helpdesk (facilitating conditions), overweeg om iets (in het begin) verplicht te maken. Je kan alle factoren uit de modellen meenemen, bewaken en gebruiken als instrumenten om te komen tot gebruik en benefit van het informatiesysteem.

**Chapter 6**

# Les 6: 27/02/2015

## 6.1 Pop quiz

Zie foto’s voor vragen & opties.

Vraag 1: zie Figuur 6.1 Welke uitspraak is fout? Er zijn 3 soorten assets: human, technology en relationship. De dimensies zijn stategisch tactisch & operationeel. B is dus fout! Operationeel betekent het meten van het dagelijkse werk. → Meten van productiviteit. Het is niet tactisch & strategisch. Vraag 2: zie Figuur 6.2 D is fout!

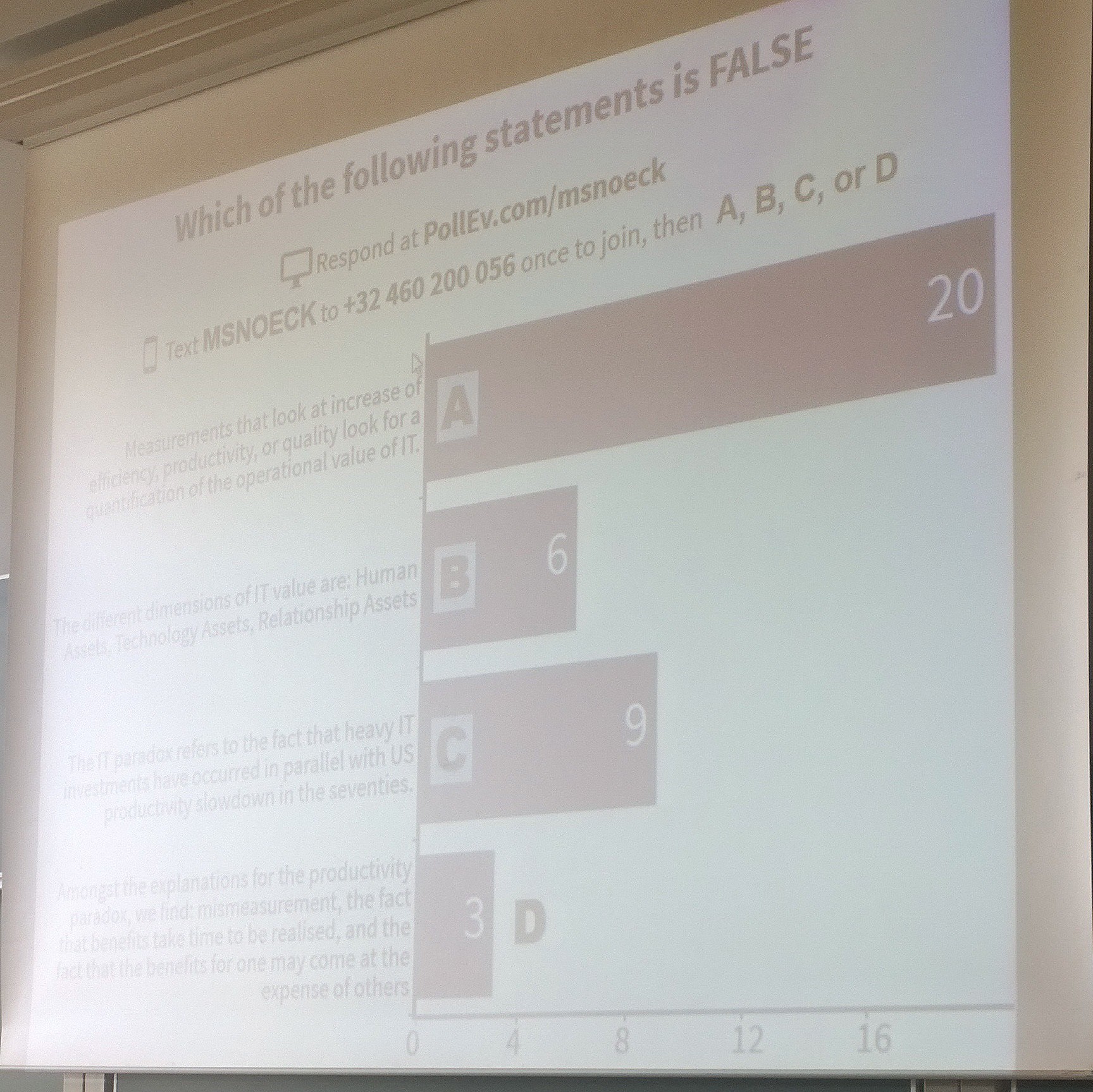


Figure 6.1: Vraag 1

Zorg dat je alle kenmerken goed kan ordenen! Je moet het niet allemaal

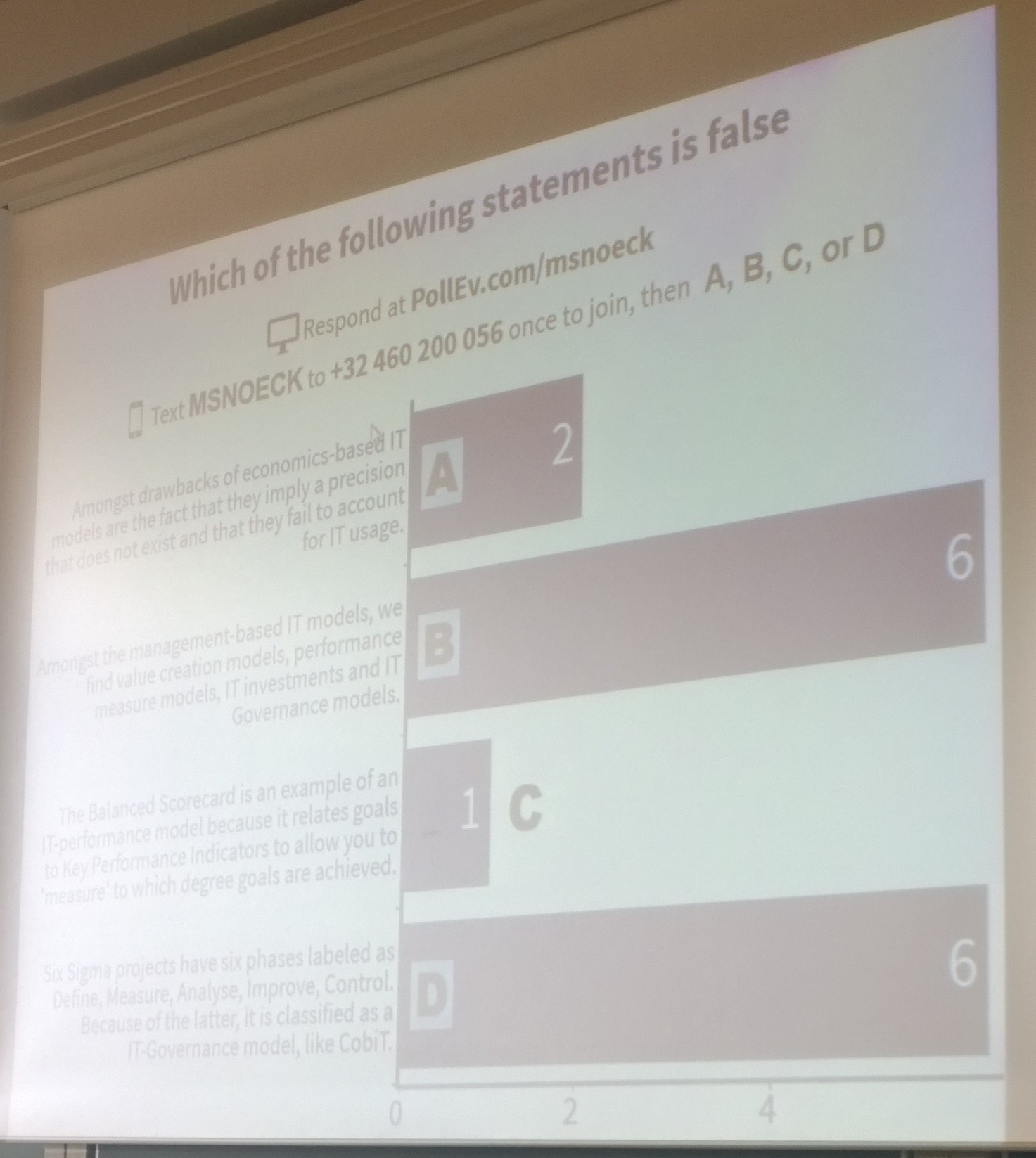


Figure 6.2: Vraag 2

vanbuiten leren, maar voldoende in detail kennen om op multiple choice te kunnen antwoorden.

## 6.2 Slides: 2B. EnterpriseArchitecture

**6.2.1 Why?**

**Slide 4:** 3 delen: Why, what, how?

**Slide 5:** Bedrijven zijn altijd op zoek naar agility, leeft daar enorm. Men botst constant op het feit dat wanneer men iets nieuws wil doen en men er technologie voor nodig heeft, het zo lang duurt eer het gerealiseerd is. In de bedrijfswereld: mergers en acquisitions, na die operatie moeten die bedrijven opereren als 1 eenheid. Daar krijg je een probleem op vlak van IT-support. Er zijn dus veel factoren die een drempel vormen en waardoor bedrijven zich niet/moeilijk kunnen aanpassen.

• Lack of understanding: aanpassen aan nieuwe omgeving gaat moeilijk als je niet weet wat er moet veranderen, maar daarvoor moet je weten hoe het vandaag werkt en hoe het nu in elkaar zit. Bij gebrek aan kennis weet je dus niet hoe je een verandering moet aanpakken & doorvoeren.

Silo’s: soort van Chinese muur tussen verschillende afdelingen. Elke afdeling werkt op zichzelf als cocon en kijkt niet over de muur en werkt niet samen met mensen van een ander departement. → Moeilijk om veranderingen door te voeren want wederzijdse impacten zijn moeilijk te begrijpen en accepteren. Virtualisatie vraagt om ge¨ıntegreerde aanpak. Met silo’s moet je dus eerst al een integratieoefening doen.

**Slide 6:** In de value chain treedt er enorm veel verandering op en de manier waarop je waarde gaat toevoegen is veranderd doorheen de tijd (vroeger fysieke bedrijven en fysieke producten, maar door virtualisatie gaat men ook services bieden. Het gaat niet alleen meer om het maken en verkopen van product, maar om een mix van producten en diensten).

**Slide 7:** Een koffieautomaat is niet meer gewoon een koffieautomaat, men is er heel wat diensten bij gaan aanbieden. Nespresso biedt niet alleen een machine, maar ook services: bestellen, delivery, customer care, recycling. Ze omringen de koffiemachine met hele hoop services, maar dat vraagt ge¨ıntegreerde benadering, ook IT-gewijs want je moet dat gaan ondersteunen. Het is veel complexer geworden dan gewoon koffie & machine verkopen.

**Slide 8:** Heel veel regulering en er wordt heel fel gekeken naar wat bedrijven doen. Je werkt in een soort van publieke ruimte waar constant gecontroleerd wordt of je wel goed bezig bent. Men gaat niet meer kijken naar individuele departementen of die goed werken, maar naar hele organisaties, inclusief de partners.

Faculteit heeft Equis-acquisitatie verworven, die kijkt naar de faculteit in het geheel. Neemt 1 of 2 opleidingen onder de loep, maar heel het departement krijgt de onderscheiding. Wanneer je bv de ondersteuning van 1 opleiding zou optimaliseren, moet je dat ineens doen voor heel de universiteit. Je moet constant aantonen dat wat je doet werkt in de hele onderneming en niet in 1 ge¨ısoleerd stukje. Ook de subcontracters en de partners. Krantenknipsels: bedrijven krijgen problemen omdat hun leveranciers niet voldoen aan bepaalde kwaliteitsnormen/ethische normen.

**Slide 9:** Klassiekere challenges: competitief voordeel halen eventueel met technologie als hefboom. Wanneer je technologie inzet, moet je daar ofwel zelf excellent in zijn, ofwel een partner zoeken die zelf excellent is.

Voorbeeld: alles is meer ge¨ıntegreerd dan vroeger. Bv nieuwe vaatwasser, via de website doorgeven dat je een probleem hebt,er wordt al dan niet contact opgenomen via email, afspraak met iemand die een volledig kantoortje meeheeft (op laptop alles registreren hoe lang gewerkt,... kleine Bancontact & printertje voor factuur). ⇒ Operational excellence: zorgen dat alles op 1 ogenblik gebeurt en het achteraf ingeven van zaken is niet meer nodig.

**Slide 10:** Wat maakt transformatie nog moeilijk? Er zijn heel veel mensen bij betrokken! Bij een tranformatie heb je

* de mensen die vragen voor de transformatie: sponsers, aandeelhouders, leiders,... → willen dat het bedrijf een bepaalde richting uitgaat,

mensen die nieuwe systemen moeten organiseren,

* mensen die het moeten ondergaan (gebruikers).

Voorbeeld van KUL: alle IT-systemen opleggen aan hogescholen, men heeft zich niet gedragen als een merger, maar als acquisition: KUL heeft zijn systeem (in overleg) opgelegd aan KHL → alle IT-systemen pushen naar alle hogescholen ⇒ een hoop veranderingen IT-gewijs, maar ook administratie, logistieke diensten, ondersteuning van onderwijs,... Owv complexiteit: bekijken als een ge¨ıntegreerd deel, niet als verschillende onderdeeltjes. Is een heel radarwerk. Je kan wel eens een tandwiel apart bekijken, maar dat volstaat niet, je moet ook kijken hoe het in het geheel past en hoe 1 deeltje een impact heeft op de rest van de organisatie.

**Slide 11:** Voor KUL: uitbreiding buiten Leuven: als je nu kijkt naar het programmaboek van de KUL: opleidingen per stad, software moet ook aangepast worden, ook reglementen → facultaire aanvullingen (per campus moeten aftoetsen). ⇒ Impact op alles.

**Slide 12:** Hoe pakt men het aan indien men geen enterprise architectuuraanpak heeft? Werken met strategische ontwikkeling → hoe in de praktijk uitwerken en projecten defini¨eren met change management en resources? Traditioneel: missie doel,... op de projecten ga je portfolio management en program management doen. → Hoe opdelen, prioritiseren, managen,...

**Slide 13:** Problemen met traditionele aanpak: Middelen toewijzen maar de integratie overheen de projecten is een aparte concern, iets dat je apart in het oog moet houden. Aspect van integratie is wat ontbreekt in program & portfolio management. Zit er wel in, maar niet expliciet. In vb: men was vergeten de planning na te gaan.

**Slide 14:** Terugkoppelen naar raamwerk van H&V: strategic fit, het operationele moet in lijn liggen met strategie, en ook functional fit: business & IT moeten ook met elkaar in lijn liggen. Is onvoldoende om te komen tot echt ge¨ıntegreerde benadering. Hoe ga je de strategic & functional fit waarborgen?

**Slide 15:** Extended framework: verschil tussen echte diensten die je nodig hebt & IT (bv vervroegde examenplanning moet zichtbaar zijn via KULoket: bepaalde services aanbieden. Hoe je die concreet implementeert is een beslissing in een andere fase). Je hebt 3 kolommen: business, IS (welke diensten wil je aanbieden op vlak van info), technologie. Horizontaal: om van strategie naar operations te gaan moet je eerst nadenken over hoe je je gaat organiseren: 3 rijen: richten (strategie bepalen, richt je naar bepaald doel), inrichten (structuur uittekenen van hoe je wil werken), feitelijk doel verrichten. Inrichten is van cruciaal belang om ervoor te zorgen dat wat je doet in lijn is met je strategie. Dit raamwerk illustreert de dingen beter dan dat van H&V. voor H&V is business operations zowel inrichten als verrichten. IT voor hen is zowel IS als technologie.

• Enterprise architecture: nadenken over hoe je wilt werken en gaat organiseren om op de best mogelijke manier tegemoet te komen aan de gevraagden.

Operations: feitelijk uitvoeren van transformatie: change management plan, manier van werken, wie gaat wat doen?

**Slide 16:** Kijken naar projecten: uit combinatie zijn aantal projecten af te leiden met elk eigen requirements die je een voor een kan uitvoeren. Vraag is hoe je ervoor zorgt dat elk project individueel, maar ook samen zorgt voor goede software in zijn geheel.

**Slide 17:** Vb een huis kan niet in 1 keer gebouwd worden wegens te weinig geld. Opdelen in kleine projecten die elk tot een ander team worden toegewezen. Elk team krijgt een budget. Je neemt de beste (binnenhuis)architecten en ze realiseren individueel projecten. Wat is de kans dat elk van die projectjes samen 1 mooi groot huis gaat vormen? Onmogelijk: allemaal een ander idee van waar vensters, muren, leidingen moeten komen! Individuele projecten gaan nooit netjes op elkaar aansluiten, wordt een complete ramp. Wat je nodig hebt is een common language between business and IT. Maar vooral individual projects need to be aligned. Hoe doe je dat? → Je maakt een globaal plan.

**Slide 18:** Als je probeert om de verschillende dingen op elkaar aan te sluiten, heb je een probleem. Je moet dus denken vanuit een globaal perspectief.

**Slide 19:** Doel van EA: organisatie organiseren. Je hebt je strategie en program management (lijstje van wat je wilt doen) en je moet iets toevoegen zodat ze niet overlappen. Waar ze overlappen, aligneren, zorgen dat er geen gaten zijn en dat ze samen iets moois vormen dat toegevoegde waarde heeft.

**6.2.2 What?**

**Slide 21:** Wat wil EA doen? Middelste rij bewaken, maar in het bijzonder connecties tussen die verschillende aspecten zodat de organisatie in lijn is met de strategie en een goede manier van werken oplevert voor informations niveau (denk ik) ⇒ vooral connecties bewaken.

**Slide 22:** Strategie en je wil van A naar B: verschillende routes die je kan volgen waarbij sommige routes beter zijn dan anderen (hangt ook af van vervoersmiddel, of je onderweg nog moet stoppen ergens,...).

**Slide 23:** GS: kijkt of systeem in lijn is met regels en strategie: governance van operationeel systeem op bedrijfsniveau. Onderste helft: enterprise transformation: transformation processes & governance ervan → uitvoeren en bewaken van transformatie.

**Slide 24:** EA moet inzicht bieden in waar je nu staat, future state (Waar wil ik naartoe?), hoe goed ben ik vandaag (tranformatie moet verbetering opleveren op verschillende vlakken)?, future performance (niet alleen waar je wil zijn, maar ook hoe goed je wil zijn) ⇒ weten in welke richting je wilt: EA biedt instrument om te weten waar je vandaag bent, hoe goed je bent, waar je wil zijn en hoe goed je wilt zijn. Daarrond hangen verschillende managementtechnieken.

**Slide 25:** Situatiebeschrijving (Waar ben ik vandaag?), richting (Waar wil ik naartoe?), gapanalyse (Welk verschil is er, hoeveel planning is er nodig?), tactische planning, operationele planning (kleine stukjes afbakenen), selection of partial solutions (met welke instrumenten?), solution architecture.

EA is een continu proces: wereld staat niet stil dus uw vlag kan verplaatst worden gedurende het project → constant herevalueren waar je naartoe wilt, hoe je het moet aanpakken en met welke oplossingen je er kan geraken.

**Slide 26:** Wat nodig op vlak van instrumenten? Sociotechnical: niet alleen technologie, maar ook mensen, werkprocedures, organisaties. Ook business architecturen, resources, mensen en hun doelstellingen. Elk woord in de definitie is belangrijk!!! Perspectieven, types instrumenten:

* Regulations: regels, als bedrijf maak je afspraken rond regels die je wil toegepast zien. Bv als je een huis gaat bouwen kan je zeggen dat het toegankelijk moet zijn voor mindervaliden dus moeten alle deuren minstens 90cm breed zijn, geen dorpels,... → Normen en standaarden opleggen. Ook technische standaarden: open source, of net niet? Rules en guidelines door omgeving opgelegd zitten hier niet bij in! → Interne afspraken die je als bedrijf maakt.
* Design: plan (figuur rechts), description aspect.
* Patterns: verzameling van best practices. (Deel)oplossingen waarvan je weet dat ze goed werken.

**Slide 27:** EA gaat u enkele dingen bieden (zie definitie op slide 27). Descriptions omvatten de verschillende aspecten gegeven.

**Slide 28 ev:** Verschillende soorten EA frameworks. Verschillende categorie¨en:

* Design: helpen om beschrijvingen op te stellen: verschillende vormen van abstractie, geen details over dikte muren, hoogte deur,... maar wel verschillende perspectieven om complexe systeem toe te lichten. Je gaat het wel moeten opsplitsen voor loodgieter,...
* Regulations: inherente wetten, opgelegde wetten, wie moet wat doen en wanneer? → Meer het governance perspectief.
* Patterns: vooral kijken naar boeken en websites: catalogen waarin je deeloplossingen kan gaan zoeken over hoe je het concreet moet aanpakken.

Video

**6.2.3 How?**

**Slide 33:**

* Voorbeeld regulations perspectief: TOGAF: raamwerk dat uitgebreid beschrijft hoe je architectuur aanpakt, wie betrokken is en welke taken moeten gebeuren. Hele cataloog aan handleidingen.
* Design focus: raamwerk van ZACHMAN.
* Design focus: architectuurraamwerk: ARCHIMATE als aanvulling van ZACHMAN.

**Slide 34:** TOGAF: heeft 2 componenten: architecture development methods: hoe ontwikkel je architectuur met bijhorende richtlijnen & enterprise continuum (handleiding met best practices)?

Methodes worden beschreven en de stappen die doorlopen moeten worden en de perspectieven die je moet hanteren. Schema:

1. Je begint bovenaan in het groen (wanneer je oefening begint, eerst concrete afspraken maken binnen bedrijven over bv hoeveel middelen je mag gebruiken voor de oefeningen en wat de doelstelling is van de oefening op zich). → Randvoorwaarden van het project eerst afbakenen.
2. Architecture vision: waar willen wij eindigen: vlag planten.
3. Business architecture: hoe je het systeem wil veranderen, aanpakken. Kijken wat de plannen betekenen ten opzichte van de huidige manier van werken. Ook kijken naar de infrastructuur van het bedrijf en die eventueel herbekijken.
4. Information system architecture: gegeven onze business, datgene waar we daarnaartoe willen, wat hebben we dan nodig op vlak van informatiesystemen? → Kolom 2 in dat raamwerk van daarnet. Vooral vanuit service perspectief. Kijk naar welke data & gegevens er nodig zijn en welke toepassingen nodig zijn.
5. Technologie architectuur: wat hebben we vandaag, wat willen we morgen hebben?
6. Opportunities & solutions: welke routes, mogelijkheden, zijn er? Welke oplossingen zien we? → Selectie maken hoe je denkt van A naar B te geraken.
7. Migration planning: plannen voor het uitvoeren van de transitie van A naar B.
8. Implementation governance: bewaken of je op de juiste weg bent.
9. Architecture change management: op een gegeven moment zal iemand zeggen dat de strategie aanpassingen nodig heeft: vlag die geplant is moet verplaatst worden. → Wie kan zo’n voorstel formuleren? Wie beslist over het verzetten van de vlag?

In de loop van al die fasen en stappen ga je allerhande dingen opschrijven, in de requirements management. TOGAF geeft daar weinig instructies over hoe je dat moet neerschrijven en bijhouden. TOGAF gaat over de cirkel. ⇒ Continuous change!

**Slide 35:**

* Preliminary phase: je gaat moeten afbakenen waar de oefening betrekking op heeft: nationaal, internationaal, wat is inbegrepen, wat niet? Eventueel met federated architectures: oefening herhalen voor verschillende locaties en daarna pas integreren.
* Architectuurdomeinen: business, data, applications, technology: beseffen dat het niet alleen over technologie gaat, maar ook over het bedrijf.
* Level of detail: als je een hoog niveau van detail wilt: ZACHMAN, ander raamwerk.
* Time horizon: kijken naar wat je wanneer gaat beschrijven en hoe je gaat werken met tussentijdse architectuur.

**Slide 36:** Fase B: Business Modelling aan de hand van business process models. Verschillende mogelijke manieren om bedrijfsdingen neer te schrijven.

**Slide 37:** Class models: daarin beschrijf je belangrijke data-objecten en hun relaties.

**Slide 38:** ZACHMAN: gaat over het middelste luik van TOGAF. TOGAF beschrijft hoe je de buitenste bollen moet uitvoeren, maar dus niet goed hoe middelste bol moet gebeuren. ZACHMAN geeft een breder en vollediger perspectief over modellen die je kan maken. Inspireert zich op het bouwen van huizen. Verticaal: aspecten: wat(info), how, where, who, when, why. Why is link met strategie.

Niveaus van abstractie: rijen: scope, enterprise (punt B bij TOGAF), IS functionality (hoe business ondersteunen door IS), design, subcontractor, code. In deze cursus zijn vooral de eerste 3 dingen interessant.

**Slide 39:** Elke kolom heeft een bepaald generisch perspectief. Gaat ervan uit dat al die dingen onafhankelijk zijn, wat in de praktijk niet altijd zo is, wel relaties tussen de aspecten.

* Wat zijn de dingen? Maar ook relaties!
* Processen: relaties tussen processen.
* Where: node-line-node: communicatiepaden, elementen in netwerk → verbanden ertussen.
* People: wie werkt met wie?
* Event
* End-means-end: waarom doen we bepaalde dingen?

**Slide 40:** Heel belangrijk: onderscheid tussen demand (gebruiker, sponser, owner, klant) en supply (aannemer, bouwer). De architect vertaalt wat de gebruiker wil naar de aannemer toe. Denk in termen van het bouwbedrijf. De architect is degene die de gebruiker bijstaat om ervoor te zorgen dat wat er gebouwd wordt voldoet aan de noden van de gebruiker. Wettelijk ook: architect mag nooit tot het bedrijf horen van aannemer. Architect staat aan de kant van de gebruikers. In de softwarewereld is dat niet zo gescheiden: softwarebedrijven die zowel architect als aannemer zijn → geen goed idee! ⇒ Als gebruiker, bouwheer moet je de architect aan jouw zijde hebben.

**Slide 41:**

* Scope: wat neem je mee, wat niet?
* Owners view: business aspecten. Belangrijk dat rij 2 agnostisch is van IS. Geen rekening houden met het bestaan van IS.
* Gewenste IS-diensten die je zou willen krijgen. Wanneer je software gaat kopen, kan je rij 3 interpreteren als IS-diensten die er al zijn. Matching maken tussen supply en demand (rij 3 zit op de grens).
* Rij 4: technologiemodel: beschrijven hoe het systeem in elkaar zit.

**Slide 42:** Samengevat: owner is verantwoordelijk voor het beschrijven van business architectuur → business IT alignment! → Middelste rij en kolom business: organizing the organisation.

**Slide 43:** Wat heb je nodig?

**Slide 44:** Constructiebedrijf: degenen die de software gaan schrijven en de technologische realisatie van het project.

**Chapter 7**

# Les 7: 02/03/2015

## 7.1 Slides: 2B. EnterpriseArchitecture

**Slide 40:** Beschrijving eerste 3 rijen is zowat het belangrijkste in deze cursus.

**Rij 1:** Scope (belangrijkste): afbakening van wat we meenemen in de architectuuroef en wat niet.

**Rij 2:** Uittekenen business perspectief: Hoe ziet ons bedrijf eruit? Wat is er belangrijk?,... → IS-agnostisch.

**Rij 3:** Wat van die taken en procedures wordt ondersteund door IS’en? Wat is geautomatiseerd en wat niet. Al bestaande software: zegt wat al bestaande software voor het bedrijf kan betekenen.

⇒ Requirements ten aanzien van hele reeks van projecten die samen de strategie van bedrijf moeten waarmaken.

Rij 4, 5,6: technisch ontwerp, bouwen, kopen van software.

**Slide 43:** Architect moet aan de demand-side staan, niet aan de kant van de supplier.

**Slide 45:** Scope (eerste 2 zijn het belangrijkste):

**Datadimensie:** Welke elementen zijn in/uit de scope? Iedereen die ingeschreven is aan unief: in scope. Mensen die niet ingeschreven zijn, maar het wel overwegen: heel lang uit scope (registreerden geen info over mensen die kwamen kijken), over de jaren heen is men gaan beseffen dat men ook de lead (potenti¨ele klanten) moet registreren. → Hoeveel mensen die zijn komen kijken hebben zich daarna ook effectief ingeschreven? ⇒ Meer vragen beantwoorden vanuit businessperspectief. Scope-afbakening: duidelijk zeggen wat binnen en buiten de scope zit, maar er is een grijze zone (dus bv mensen die zich komen informeren). Zo kan je voor verschillende zaken/elementen kijken of je die meeneemt of niet. Belangrijk om te beslissen wat software allemaal moet kunnen.

**Functie:** Organiseren van infodag: zit dat mee in het studentendomein of niet? Werkprocessen & taken, welke daarvan nemen we mee in onze oefening en welke laten we erbuiten?

**Netwerk:** Locaties waarover het bedrijf beschikt. Je kan beslissen om bepaalde zaken alleen voor bepaalde locaties beschikbaar te stellen.

**People:** Mensen die taken uitvoeren. Zit bijna impliciet in functiedimensie.

**Time:** Het gaat over events, gebeurtenissen, worden vaak ook gecapteerd door bedrijfsprocesmodellen.

**Motivatie:** Maakt link met strategie. Als je bv zegt dat je de buitenlandse studenten die appliceren meeneemt in systeem, moet je dus motiveren waarom. → Beslissing om iets in/uit scope te zetten linken met bedrijfsstrategie.

**Slide 46:** Rij 2: ownerperspectief: bij huis: wat is belangrijk voor de klant? Wat betekent wonen voor u, wat is uw bedrijf, waarover gaat het? Wat zijn uw basisproducten, werkprocessen,...? In rij 1 enkel oplijsten, in rij 2 uittekenen en defini¨eren.

**Data:** Defini¨eren van wat een product is. Is op zich niet relevant: van wanneer is iets een product? Product is niet alleen alles wat commerci¨eel verkoopbaar is. Je moet defini¨eren wat je bedoelt als je spreekt over je product. Vocabularium defini¨eren! In sociale zekerheid: over apothekers: wat is een apotheker? Iemand die een diploma heeft van apotheker maar misschien een andere functie uitoefent? Iemand die eigenaar is van een apotheek (al dan niet zonder diploma), iemand die apotheek uitbaat in ziekenhuis, is die apotheker? ⇒ Wat bedoelen we als we ’apotheker’ zeggen? → Defini¨eren wat je bedoelt gebeurt op rij 2!

**Functie:** Gelijkaardig: beschrijven van werkprocessen, hoe je dingen doet. Bv heel ISP-goedkeuringsproces valt onder functiedimensie.

**Netwerk:** Je kan gaan uittekenen zoals in rij 1.

**People:** Organigram wie waar werkt in welk departement.

**Time:** Timing weergeven van zaken.

**Motivation:** Link maken met business plan: KPI’s.

**Slide 47:** Rij 3: link maken naar IS-diensten.

* Beschrijven wat van rij 2 moet ondersteund worden in het IS en hoe je dat gaat doen. Je gaat komen tot een datamodel.
* Functie: abstracte beshcrijvingen van bedrijfsprocessen: omvormen tot executable business processes.
* Analoog voor network, people, time, motivation. ⇒ Alles meer beschrijven vanuit IS, technisch perspectief.

**Slide 48:** Rij 4,5,6: gedetailleerd van hoe de code in elkaar zit (4&5) & systeem in uitvoering (rij 6).

**Slide 49 ev:** Bibliotheekvoorbeeld. Limo:

Rij 1/What: Oplijsten & defini¨eren van belangrijke concepten: wat is het uitleenbare materiaal in de bib? Multimediamateriaal is niet uitleenbaar, maar wordt wel gecatalogeerd. Tijdschriften ook catalogeren, maar de papieren tijdschriften zijn ook niet uitleenbaar,... Defini¨eren wie de mensen zijn die in-scope zijn in het Limo-systeem: student & werknemer, alumni, derde partijen,...→ Ook opschrijven welke categorie¨en van mensen in aanmerking komen.

Rij 2/What: Bedrijfsmodel maken: ER-model: je hebt leden, bibliotheken en items (dus dat zijn de 3 rechthoeken, de objecten) en dan heb je de relaties.

Rij 1/How: Lijst van processen: process map waarbij je belangrijke domeinen gaat afbakenen en collecties van processen gaat noteren. Je lijst alle processen op en catalogeert ze naar core processen, ondersteunende processen,... in uw bedrijf. In ons vb: alles wat met lenen, reservaties,... te maken heeft.

Rij 2: BPM’s.

Rij 1/Where: Meegeven welke bibs meegerekend , lijst van locaties.

Rij 2/Where: Locaties met elkaar linken. Relatie tussen bepaalde bibliotheken die onder andere bibliotheken vallen. → Netwerk van bibliotheken met hi¨erarchie.

Rij 2/Who: Organigram: wie zit waar in welke afdeling, hoe zijn die met elkaar verbonden?

Rij 2/Who + How: Je kan bijkomend uw people dimension linken aan uw processen. → Combinatie van persoon & hoe → je neemt de verantwoordelijkheden mee. Bij BMPn (**slide 54)** zie je dat de how-dimensie al linkt met who-dimensie.

Rij 1/Why : Motivatie: Link met verschillende aspecten van strategie mbt onderzoek, onderwijs, beheren collectie. Het is de bedoeling dat je de verschillende elementen uit rij linkt met strategie.

Rij 2/Why: Strategie om visie te realiseren.

**Slide 60:** Bib is afgebakend domein, is maar een klein deeltje van KUL, dus vraag is wat er gebeurt als je al die schema’s moet maken voor heel de KUL. Typisch gaat men die architectuur benaderen vanuit verschillende invalshoeken en zo deeltjes afbakenen, deelarchitecturen. Meestal: functionele domeinen afbakenen: alles wat met onderzoek te maken heeft, onderwijs, logistiek,... afbakenen. Daarbinnen kan men binnen bepaalde domeinen nog afbakeningen doen. Je mag niet vervallen in het probleem van de silo’s waarbij elk domein een eigen silo vormt met Chinese muur rondom zich zonder verbinding met andere domeinen. Want bv bib zit tussen onderwijs en onderzoek. Niet zomaar in 1 hokje steken en niet toelaten in een ander. ⇒ Federated architectures: op hoger niveau abstracte modellen maken, lijst van concepten. Op dat niveau de splitsing maken en identificeren wat op de rand zit tussen 2 domeinen.

**Side 61:** Andere manier: werken per aspect. Eerst data-architectuur, daarna procesarchitectuur. → Nadeel: als je die apart gaat maken, ga je bv bij het procesluik data nodig hebben dus moet je verwijzen naar de data. Binding tussen verschillende deelaspecten is zo groot dat het moeilijker realiseerbaar is. Het vorige (**Slide 61**) is dus beter af te bakenen en dus de favoriet van de prof.

**Slide 62:** Anemoonproject van KUL (voor 2000): eigen software van KUL. In jaren 70: stockeerde datum met alleen 2 cijfers voor jaar (dus voor 1900). Men was ervan overtuigd dat het tegen 2000 al lang vervangen ging zijn. Software was niet op voorzien op het gebruik van datums vanaf 2000, maar was initi¨eel geen probleem. Men had geen tijd om de software te vervangen, tegen 2000 had men nog geen nieuwe software. Vanaf dan draait dus alles in de soep. Vervanging van Frank door Euro: alle toepassingen moesten ook hiernaar aangepast worden. → Gigantisch onderhoudsprobleem van alle bestaande software. Beslist om te starten met architectuuroefening en nadenken hoe we de organisatie zien en welke ondersteuning we willen en hoe we aan de software gaan geraken. Besloten om software te gaan kopen bij SAP (mits aanpassingen). KUL is zeer groot dus gewerkt met federated architecture: vandaar anemoon: bloem met 6 blaadjes. Staan niet los van elkaar, volgorde waarin men het heeft uitgewerkt: eerst zorgen dat luik financi¨en in orde is (want voor heel veel zaken geld nodig). → Eerst financi¨en doen, dan pas personeel. Education: men moet de prof kennen die onderwijsopdracht heeft, maar dan ook in welk lokaal het gaat doorgaan → gaat gebruik maken van logistics & technics (eerst) en daarna personnel. Research: finance en personeel: stond redelijk los. Studenten: kwamen laatst want bouwen verder op education. Er zit een logica in, je kan dingen afbakenen, maar er zit een logica in de volgorde waarin elke architectuur uitgewerkt gaat worden.

**Slide 63:** Anemoonproject: hoe in de praktijk?

* Architectuurcel waarin representatieve gebruikers van de domeinen zitten: domeinexperten. Men flankeert die met een informatiearchitect → ownerrol en rol van architect.
* Informatiearchitect maakt brug tussen gebruiker en aannemer. Architect is ook verantwoordelijk voor het stroomlijnen overheen verschillende domeinen. Het is de verantwoordelijkheid van de architect dat wanneer iemand ingeschreven is als doctoraatsstudent in educationdomein, deze ook doorstroomt naar onderzoeksdomein en vice versa. Moet onderhandelen tussen gebruikers & aannemers.
* Aannemers: ICTS domain responsible: SAP is niet plain vanilla overgenomen, is gecustomiseerd. KUL heeft binnen ICTS een hele ploeg mensen die vertrekken van SAP en daar software voor bijschrijven. In ICTS-domein heb je een verantwoordelijke per domein. ICTS is aannemer die gebruikmaakt van prefab die door SAP wordt geleverd.

Integratie: co¨ordinatiecel waar alle informatiearchitecten en ICTS-verantwoordelijken samenzitten

**Slide 64:** Architectuur met gebruikers & informatiearchitecten: rij 1 & 2. Architect vertaalt het businessmodel naar IS-model en ICTS zet het om naar bruikbaar model.

**Slide 65:** Problemen: op zich een handig kader om een aantal zaken te beschrijven, maar als je het tot in detail zou gaan toepassen: problemen.

* 36 cellen: veel te veel modellen nodig.
* Consistentiebewaking: als je in het procesmodel schrijft dat je bepaalde zaken nodig hebt, moet dat in de wat-kolom komen. Als je bepaalde mensen iets gaat laten uitwerken: in hoe-kolom. Zachman biedt geen info over wat waar exact moet komen.
* Vergt heel veel documentatie.
* Zegt niet hoe je specifiek moet werken. Je bent nog niet helemaal klaar met alleen maar Zachman. Je hebt dan wel een denkkader, maar er zijn nog heel veel extra zaken die uitgewerkt moeten worden.
* Tool support: is er deels. Er zijn een aantal tools die verderwerken op Zachman en technieken aanbieden en consistentie aanbieden.
* Continuous: je zou heel de tijd documentatie enzovoort moeten aanpassen!
* Fel met elkaar verbonden: niet altijd duidelijk waar je wat moet schrijven. Dus vaak in de praktijk enkel what en how en de rest een beetje aan de kant. Als je dan focust op eerste 3 rijen: nog maar 6 cellen → veel hanteerbaarder. Verkleinen probleem van documentatie, niet-hanteerbaarheid van raamwerk → veel makkelijker en toepasbaarder.

**Slide 67:** Kanttekening: impliciet zit ook wie & wanneer erin! Vraag 1 hierover: zie Figuur **??** Rij 3 heeft met software te maken, dus is hier niet van toepassing! Statement defini¨eert wat met full-time program wordt bedoeld en geeft er betekenis aan. Bij scope: dit wel en dat niet. Hierin zit scope er wat in, maar op het moment dat je de definitie neerschrijft is scope-afbakening al achter de rug. Als je statement zou hebben van geen rekening te houden met deeltijdse studenten: scope. Statement is business rule: minimaal 54, maximum 72 punten. Juiste antwoord: behoort tot rij 2!

Vraag 2: zie Figuur 7.2 Uitspraak bevat 2 zinnen: wanneer je vrijstelling krijgt wanneer je minstens 10/20 haalt → businessrule, softwareagnostisch. Heeft niks te maken met IS. Die zin behoort tot rij 2, defini¨eert wat een vrijstelling is. Tweede zin: ISP (SW) moet vrijstellingen automatisch toekennen: service die je verwacht van het systeem. Dat behoort tot rij 3.

Vraag 3: zie Figuur 7.3 Het gaat over het ISP-systeem: software, dus rij 3 heeft er zeker mee te maken. A is sowieso fout! Er staat weinig over businessrules, dus B is ook uitgesloten, want geen regels! Er wordt aan scopeafbakening gedaan: wat nemen we mee en wat niet? Dus combinatie van rij 1 en 3. punten die ergens anders behaald zijn: buiten scope. Dus rij 1: scope, rij 2: regels, rij 3: systeem Verschillende uitspraken zijn ook mogelijk als antwoordmogelijkheden.

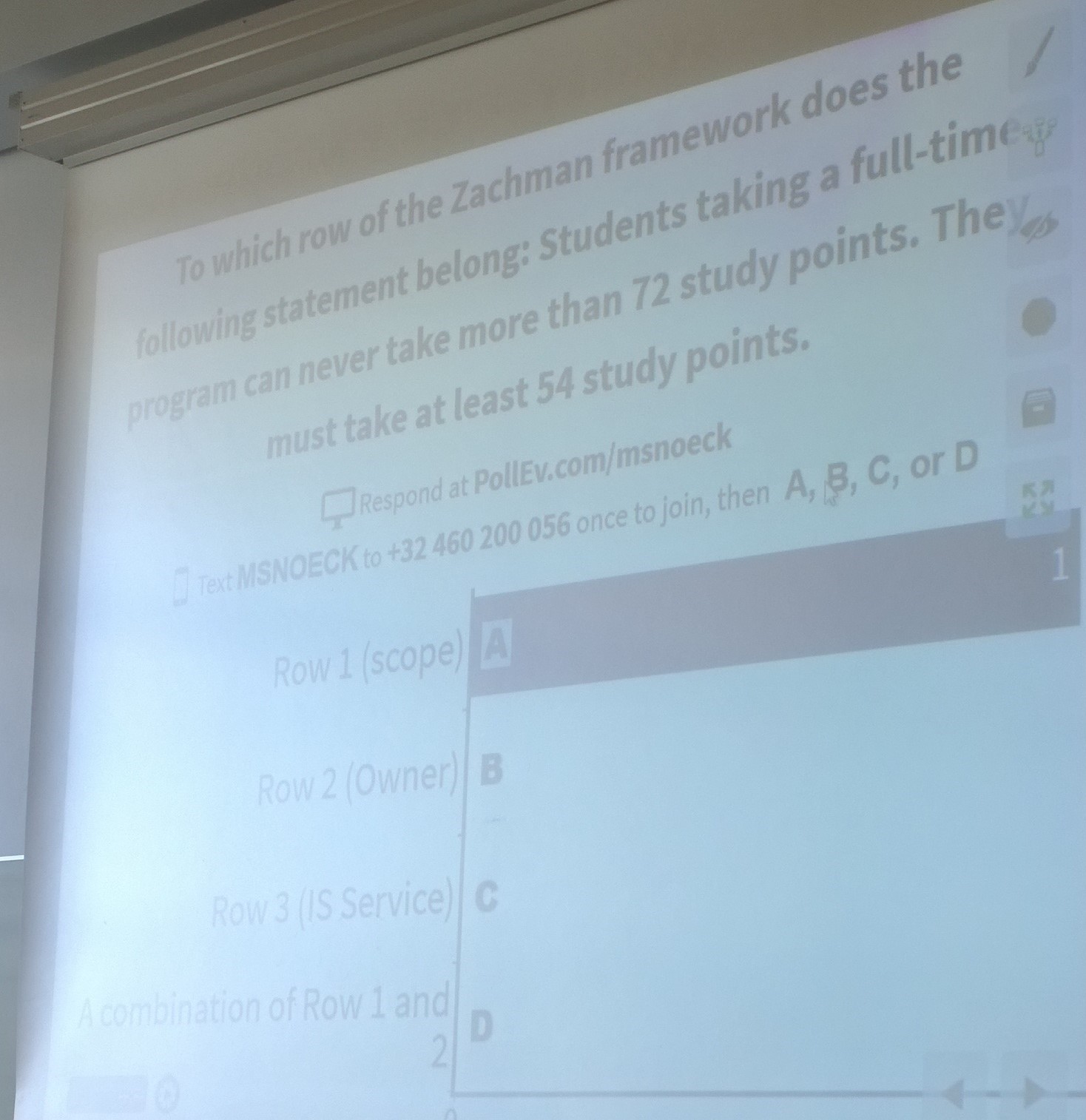


Figure 7.1: Vraag 1

## 7.2 Slides: 3A BPManagementIntro

**Slide 3:** Bedrijfsprocessen: wat is een bedrijfsproces? De definitie van een BP: verzameling van gerelateerde gebeurtenissen, activiteiten & beslissingen die een aantal actoren mee betrekken (mensen, machines, systemen → heel ruim) en de bedoeling is dat het geheel van gebeurtenissen leidt tot een meerwaarde → stukje van value chain. Meerwaarde kan zijn voor de klant, maar ook intern (voor collega of ander departement).

* Order to cash: alles afhandelen van bestellen tot levering → tevreden klanten.
* Quote to order: offerte maken, aanbod doen die hopelijk leidt tot bestelling → meerwaarde voor bedrijf.
* Procure-to-pay: analoog aan order & quote.
* Application-to-approval: bv voor Erasmus.
* Claim-to-settlement: bv schadeclaim na auto-ongeval. • Fault-to-resolution: probleem dat je opgelost wil zien.

**Slide 4:** Wasmachine werkt niet (Fault-to-resolution): als geen garantie meer: customer unhappy, geen waarde toegevoegd. Je wil zorgen dat de klant in beide gevallen (warranty en geen warranty) toch happy is. Je wil de kans vergroten

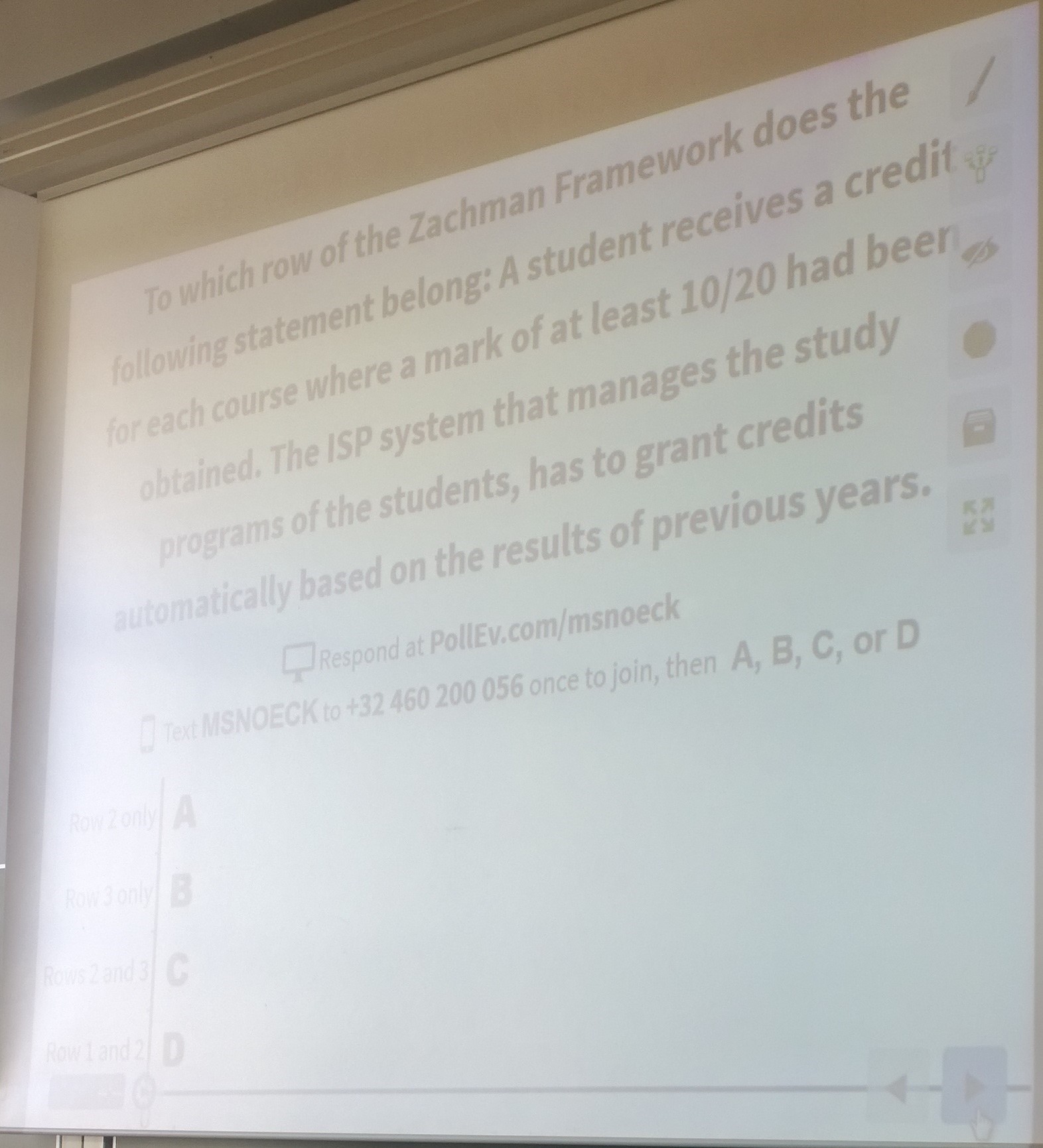


Figure 7.2: Vraag 2

op tevreden klanten. BPM onderzoekt het proces zodat je de negatieve waarden naar beneden kan halen. Mogelijkheid: warranty-check die eerst op het einde gebeurde nu in het begin te doen. Call center vraagt dus bv details over aankoop. Indien dit zo is, gratis hersteld. Anders: keuze voor herstelling of nieuwe machine (niet gratis!). Kans op tevreden klant veel groter want kost op (mislukte) reparatie is kleiner. Volgorde van stappen dus bestuderen en eventueel wisselen.

**Slide 5:** Elk proces heeft een bepaald resultaat (pos/neg). Aantal varianten op fault-to-resolution (zie slide). Varianten in dalende volgorde van waarde.

**Slide 6:** Welke elementen zitten in bedrijfsproces? Samengevat: bedrijfsproces levert een bepaald resultaat af, kan positief of negatief zijn. Positief: meestal voor de klant gewild. Gebruikt aantal actoren (klant, mensen in onderneming,...) en maakt gebruik van resources en middelen (onderdelen, lokalen,...). Bestaat uit events, activiteiten en beslissingen.

**Slide 7:** BPM = het geheel aan technieken om bedrijfsprocessen te analyseren en uit te voeren. Je gaat moeten praten met mensen om te zien hoe het bedrijfsproces werkt, uittekenen, analyseren. Als je een manier van werken hebt gevonden waar je achter staat, gaat het domein van BPM de tools geven om dat te ondersteunen.



Figure 7.3: Vraag 3

**Slide 8:** Waarom? Een computer maakt niet per definitie dingen beter. Proces automatiseren: als het goed werkt, gaat het sneller/effici¨enter gebeuren. Maar als er een fout in proces zit, gaat een computer de fout gewoon sneller en vaker maken. Belangrijk om, voordat iets geautomatiseerd wordt, te kijken of het proces goed in elkaar zit en daarna pas te automatiseren.

**Slide 9:** IS geeft geen business value, maar geeft de mogelijkheid om de manier van werken te veranderen. Kijken HOE je het gaat aanpakken, niet gewoon het proces klakkeloos overnemen.

**Slide 10:** Elementen die erbij betrokken zijn: het gaat niet alleen om ITsystemen, maar ook over klanten, mensen binnen de organisatie, partners, leveranciers en meer technische zaken zoals IT-infrastructuur en gegevens.

**Slide 10:** Doorlopen van bepaalde een cyclus om doel te bereiken:

* Proces identifications (zie process map daarnet als voorbeeld library): process map komt hier terug: identificeren welke processen in en uit scope zijn, welke processen interessant zijn om onder de loep te nemen.
* Process discovery: Specifieke, individuele processen ontdekken: praten met mensen, documentatie inspecteren, kijken hoe men in het bedrijf werkt.
* Process analysis: as-is process model wordt verkegen vanuit process discovery: base line: hoe je vandaag werkt. Op basis van dat model ga je kwalitatieve of kwantitatieve analyse doen.
* Process redesign in functie van uw vision, het proces herontwerpen.
* Process implementation: to-be process model dat je net bekomen bent gaan implementeren. De organisatie op een bepaalde manier doen werken. Instructies geven, infrastructuur & middelen aanpassen.
* Process monitoring and controlling: je krijgt een proces in uitvoering, dat kan je gaan monitoren en controleren. Bv herstelproces wasmachine hertekenen: checklist om te zorgen dat kan nagegaan worden of het probleem door de klant zelf kan opgelost worden, indien dat niet kan, dan waarborg, technieker,... → Vanaf dan meten en tellen hoe vaak het gebeurt dat de klant het probleem zelf kan oplossen (eventueel in handleiding zetten,...). Je gaat monitoring doen ⇒ conformance and performance insights → doet de call center lady wel de checklist aflopen? Kijk hoe het de performance verbetert.
* Opnieuw process discovery → terug as-is model: terug analyseren, verbeteren,... ⇒ permanent verbeteren.

**Slide 11:** Process identification: process landscape: heel abstract, verder verfijnen tot abstract process models (uittekenen): quote handling, invoice handing,... uittekenen: detailed process models. De graad van detail waarin je processen gaat beschrijven gaat dus van boven naar onder.

**Slide 12:** Op basis waarvan kiezen hoe je gaat verderwerken? Bedrijfsstrategie gaat de richting aangeven hoe je kiest,hoe je dingen onder de loep moet nemen. Kost, tijd en kwaliteit zijn mogelijke hoofdrichtingen waarin je wilt gaan [SLA violations: beloftes nakomen, in welke mate? Bv ”binnen 7 dagen antwoorden”].

**Slide 13:** Eens je gekozen hebt welke processen je wil aanpakken, ga je het proces in detail modelleren: je bekomt een process model.

**Slide 14:** Analyse maken: kwalitatief/kwantitatief.

**Kwalitatief:** Issue register, root-cause analysis (verband leggen tussen problemen), PICK charts (prioritiseren: possible, implement, challenge, kill).

**Kwantitatief:** Eerder wiskunde: wachtrijtheorie.

Zie vb **Slide 15** voor issue register. Kijken wat de assumpties zijn en impact is.

**Slide 16:** Kwantitatieve analyse: sollicitatie: verdeling van hoe ze binnenkomen, controle op volledigheid. → Cijfers erbij geven: info waarop je kan verderwerken.

**Slide 17:** Simulatie: op basis van een simulatie proberen een beeld te schetsen wat bepaalde zaken kunnen betekenen voor de onderneming.

**Slide 18:** Process-redesign: van as-is naar to-be: inspelen op de 4 assen.

**Slide 19:** Je tekent uw to-be procesmodel uit, nieuwe tekening (geen BPMN in vb), invoeren in tool, demo van geven en die tool zorgt dan dat je ITondersteuning krijgt voor je proces.

**Slide 20:** Terug **Slide 11** in de les.

**Slide 21:** Bij mij: hoe past het in het totale plaatje: links onderaan.

**Slide 22:** Waarover gaan de processen? Business goals, processen defini¨eren (identify en discover) en dan business process execution.

**Slide 23:** Band met software ontwikkeling? Al die zaken net beschreven bepalen niet alleen de strategie, maar ook de software requirements.

**Slide 25:** Koppeling met EA: in TOGAF fase 2: business architecture zitten activiteit modellen.

**Slide 26 ev:** Zachman: tweede kolom, process identification (rij 1), uittekenen (rij 2), implementeren & IT-support (rij 3,4,5,6).

**Chapter 8**

# Les 8: 06/03/2015

## 8.1 Slides: 3B BusinessProcessModellingEssentials

**Slide 2:** Wat is een proces? Bedrijfsproces uitgevoerd door actoren (mensen die participeren), die gebruiken objecten (lokalen, geld,... → resources) → resultaat afleveren, hopelijk postief. Componenten: gebeurtenissen, activiteiten en beslissingen

**Slide 3:** Onderscheid activiteiten en gebeurtenissen:

**Activiteit:** Vergt zekere tijd: start nu en is nu gedaan en daartussen verloopt bepaalde tijd. Gebonden aan gebruik van resources en er is een toestandsverandering: toestand voor & na activiteit is anders. Bv invoeren bestelling: vraagt tijd en voor & na een verschil: ervoor geen bestelling, erna wel.

**Event:** Bepaalde conditie, omstandigheid dat onmiddelijk gebeurt. Er is geen tijdsaspect aan. Bv: na invoeren bestelorder: ”Het is ingevoerd” of activiteit is gestart. → onmiddelijk. Rock Werchter bv is in BPMN-termen een activiteit en geen event! Wij bedoelen een heel specifiek iets: iets dat gebeurt, onmiddelijk is, geen tijdsduur heeft en geen resources gebruikt.

**Slide 4:**

**Business objects:** (Niet) tastbare objecten binnen de organisatie. Prof, studenten, vak, lokaal: business objects. Wij zijn ook actoren want we nemen deel aan activiteiten. We gaan ervanuit dat voor niet-tastbare objecten info wordt bijgehouden → elektronisch/virtueel object.

**Actoren:** Mensen die deelnemen aan de systemen, maar ook de systemen zoals bv Toledo.

**Slide 5:** Proces schema gaat die dingen combineren: control flow (rechthoek rechtsboven is control flow), data: die files bovenaan (invoice, report, invoice) en resources: swim lanes om aan te geven wie wat doet (horizontaal).

**Slide 6:**

* Control flow: legt link tussen activiteiten en events. Zegt wat er moet gebeuren en wanneer.
* Data: welke info hebben we nodig en welke produceren we?
* Resource: swim lanes: aanvulling op control flow door activiteiten in juiste swim lane te plaatsen.

**Slide 7:** Andere technieken zijn ook mogelijk.

**Slide 8:** Wat is PM? Gemeenschappelijke taal die je gaat gebruiken om bepaalde dingen op te schrijven: je moet met elkaar kunnen communiceren over de processen. → Afspreken bepaalde notatie zodanig dat er een eenduidig begrip is over wat het proces gaat. Laat u toe het procesperspectief met dataperspectief te linken. Je koppelt het aan resources, dus ook aan het organigram van de organisatie. → Koppeling met andere perspectieven en samenbrengen in 1 procesmodel. ⇒ Walk-through om te kijken of iedereen akkoord is, valideren,... Je kan simuleren dus ook what-if analyse mogelijk. Heel veel informaticaprojecten hebben nood aan een procesbeschrijving om te weten wat er juist ontwikkeld moet worden.

**Slide 9:** Service krijgt vorm door het proces dat aan de basis ligt van de service. Daarom moet kennis over hoe het proces verloopt ge¨extraheerd worden.

**Slide 10:** Wat is een model? En hoe zit modelleren in elkaar? Modelleren: abstractie maken van iets dat in realiteit bestaat: aantal dingen laten wegvallen (anders te complex). Vb: voor wie moet weten hoe een auto technisch in elkaar zit, maakt het niet uit wat de kleur is, de wielen,... als je aan gebruiker wilt tonen hoe de auto eruit wilt zien: modelautootje → doelpubliek is zeer belangrijk! Ook in geneeskunde: verschillende weergaven van het menselijk lichaam. ⇒ Voorstelling aanpassen aan het doelpubliek, maar ook aan wat je wilt bestuderen.

Er is geen correct/fout model, maar wel een relevant/irrelevant model. Een model kan fout zijn in de zin van: model van skelet en elleboog op plaats van knie, is uiteraard fout! Maar als je de spieren wilt bekijken, is het skelet niet boeiend.

**Slide 11:** Verschillende modellen gebouw: maquette en grondplan: grondplan nuttig voor tuinarchitect, maquette voor communicatie met toekomstige bouwheer om beeld te geven hoe gebouw eruit gaat zien.

**Slide 12:** Op het moment dat we naar proces-identificatie gaan: processen identificeren en beslissen wat ons studiedoel is: welk perspectief willen we hanteren? En belangrijk te weten welke techniek het beste is gegeven het perspectief. → Kiezen van een taal.

Terugkijken naar verschillende stappen: procesidentificatie (in welke processen zijn we ge¨ınteresseerd en welk perspectief willen we hanteren?). Aan de hand van het perspectief en het studiedoel gaan we technieken kiezen (wij gaan SIPOC en RASCII zien), om BPMN in detail te bekijken. Petrinetten: andere techniek, hier niet relevant. Als dat gedaan is, gaan we over naar Process enactment (link met automatisering en IS).

**Slide 21 ev:** Welke processen zijn in scope? We hebben een map die zegt welke domeinen er zijn in ons bedrijf, nu kijken wat we in scope nemen en welke niet.

3 verschillende studiedoelen met elk eigen publiek en notatie:

* Extern, intern, automatiseringsperspectief bv: je wil een huis kopen, dus lening aanvragen. Extern perspectief: perspectief dat een klant heeft die het bedrijf als zwarte doos ziet (kan niet zien wat intern in bedrijf gebeurt, alleen wat aan de buitenkant gebeurt), ziet hoe een lening kan aangevraagd worden, hoe snel antwoord gegeven wordt, is er ruimte voor onderhandeling? Je ziet niet wat er binnenin de bank gebeurt op het gebied van interne processen. Je weet niet waarom/hoe beslissingen genomen worden, enkel antwoord. → Hoort bij rij 2.

Intern: je trekt de doos open en kijkt naar wat er vanbinnen gebeurt: waarom geen lening gekregen? → Controle geweest op kredietwaardigheid,... Je gaat kijken naar de interne organisatie van voor welke klanten het kantoor zelf kan beslissen welke klanten een lening kunnen krijgen en voor welke het wordt doorgestuurd,... Je kijkt echt naar de interne organisatie. Het externe, de communicatie is nog steeds een onderdeel, maar je kijkt naar de werking. Als je klachten krijgt dat een antwoord krijgen te lang duurt, kan je intern gaan kijken waarom dit zo is. → Andere studiedoeleinden. → Hoort bij rij 2.

IS: heel specifiek kijken naar overgang tussen rij 2 & 3: welke taken wil ik geautomatiseerd zien, of waarvoor wil ik ondersteuning van IS? Bepaalde richtlijnen kan je buiten beschouwing laten (bv geven we klanten koekjes en koffie?). Geen juist/fout perspectief: elk perspectief heeft een eigen studiedoel en eigen doelpubliek.

* Managementniveaus (operationeel, tactisch of strategisch) operationeel: bv proces van inschrijving, goedkeuring ISP, invoeren examenmomenten. Meestal zeer duidelijk afgelijnde taken, wie dat moet doen en in welke volgorde het moet gebeuren. → Makkelijk te beschrijven aan de hand van processchema van daarnet. Lenen zich makkelijk tot automatiseren: IS-perspectief is hier heel relevant tactisch: curriculumontwikkeling, ruimte voor keuzevakken, lijst keuzevakken aanpassen,... → Processen die meer beslissingsruimte in zich houden, meer regels hebben, die wat flu-er omschreven zijn en een veel minder vast verloop hebben. Hebben deadlines die gerespecteerd moeten worden, hebben een aantal grote stappen, maar zijn niet tot in detail gemodeleerd (bv masterthesis: je moet tussentijdse presentatie doen, eindtekst indienen en verdedigen. Wat daartussen gebeurt kan je niet modelleren/regels aan geven). Procesmodel is mogelijk maar meestal heel abstract. Geen echte automatisering mogelijk zoals bij operationeel niveau management: lange termijn, strategie: bv beslissing om over te gaan van jaarsysteem naar diplomaruimte → ongestructureerde processen. Geen control flow. In het beste geval kan je daar aanduiden wie verantwoordelijk is voor wat en wie wat ondersteunt, actoren dus nog identificeren. Informatie ook, maar geen gedetailleerde stappen → heeft impact op techniek die gebruikt kan worden voor modelleren processen.
* Per functioneel domein of end-to-end (**Slide 16**) functioneel: beperken tot bepaald functioneel domein, bv binnen een organisatie. Bv: binnen unief: inplanning uurroosters probleem: bepalen van inhoud voor volgend jaar moest ingediend worden op 31 januari, maar advertenties moesten al af zijn in juni daarvoor om het pas in te voeren in september → discontinu¨ıteit. Ook bv: nu al doorgeven welke examenvorm gaat gegeven worden volgend jaar juni, maar als er iemand anders wordt toegewezen, kan dat pas in juni dit jaar (kan dus zelf niet kiezen welke examenvorm er gegeven wordt want is vastgelegd door voorganger) end-to-end: koppelt verschillende functionele domeinen aan elkaar. Vaak in overgang van ene functionele domein naar andere: fouten. Alles aan elkaar koppelen en zo kijken of het geheel vlot loopt ⇒ beschrijving, zeer nuttig om discontinu¨ıteiten te bewaken zodat de klant goede dienstverlening heeft. Je zou willen dat van bestelling tot en met levering en customer care, dat dat allemaal goed verloopt.

**Slide 26:** Keuze gemaakt: je gaat modelleren.

**Slide 27:** Control flow-gerichte technieken: leggen nadruk op volgorde waarin dingen gebeuren: modelleringstechnieken:

**Taak:** Identificieert taken en verbindt taken met pijlen.

**Toestandsgebaseerde technieken:** State diagrams voor modelleren processen: toestand zit in de bol en taak brengt object van ene toestand naar andere.

**Petrinetten:** Omvatten zowel taken als toestanden: je kan bestelling weergeven als zwart punt (token), dus 3 bestellingen: 3 zwarte punten. Taak verhuist 1 bolletje van ene cirkel naar de volgende.

**Pi-calculus**

→ Allemaal bruikbaar voor simpele processen. Rechts: andere technieken die meer declaratief zijn:

**RASCI & SIPOC:** Organisationeel: alles behalve control flow, geschikt voor managing & op strategisch niveau.

**Business rules:** Zegt gewoon puur en alleen wat de regels zijn, niet hoe ze moeten gebeuren. Gaan bepaalde volgordes afdwingen maar niet op die manier uittekenen.

→ Geschikter voor complexe en ongestructureerde processen.

**Slide 30:** RASCI: soms laat men de S vallen en dan krijg je dus RACI. Je gaat deze techniek gebruiken om aan te geven welke rol bepaalde actoren hebben in een proces. Bv geven van vrijstellingen: wie is verantwoordelijk (wie gaat op ’goedgekeurd’ klikken?), accountable: bij ons: programmadirecteur, die blijft uiteindelijk verantwoordelijk voor het al dan niet vrijgesteld worden.

Supportive: mensen aan wie je support kan vragen.

Consulteren: typisch de lesgever want stafmedewerker weet niet of het vak aan de andere unief gelijkwaardig is aan wat hier gedoceerd wordt.

Informeren: student krijgt beslissing toegestuurd en typisch ook docent zodat die weet of dat examen van student moet afgenomen worden of niet.

**Slide 31:** SIPOC: uit Six Sigma: processen uittekenen volgens input van supplier, maar kijk ook naar input, output en control flow (summier en in tekstvorm) → proces.

**Slide 32:** Vrijstelling: Supplier: student moet informatie toeleveren: welk vak voor welk vak? Student is klant. Iets meer gedetailleerd beeld dan in RASCI, maar niet stap-voor-stap proces, nog steeds vrij high-level.

**Slide 33:** Werk iteratief: RACI, dan SIPOC en indien nuttig nog iets automatisering (not sure). Globale plaatje, concrete plaatje (meer details, dus ook uitzonderingen opsommen) en dan uitvoerbaar proces.

## 8.2 Slides: 3C-BPMNBasics

**Slide 3:** Grafische notatie, maar niet alleen symbolen. Er is syntaxis (welke symbolen zijn er?), maar ook betekenis (semantiek). Regels van hoe je symbolen kan combineren met elkaar. Bv start event heeft bepaald symbool, er wordt een bijkomende regel gegeven over hoe die eruit moet zien.

**Slide 4:** Niet zuiver grafisch: 3 basissymbolen (cirkel, ruit, rechthoek met afgeronde hoeken). Per element hoort er nog allerhande extra info bij.

**Slide 5:** BPMN brengt 3 schematechnieken. Wij zien alleen process diagrams. Als je verder kijkt in handleiding BPMN: 3 types van modellen die je kan modelleren met end-to-end model: private non-executable (intern perspectief en niet geschikt voor automatisering, wel uitvoerbaar door mensen), private executable (uitvoerbaar door een computer → IS-perspectief), public (wat je wil tonen aan buitenwereld).

**Slide 6:** Belangrijkste elementen:

**Event:** Bepalen ritme van proces: start, stop even, ga verder, stop helemaal.

**Activiteiten:** Taken, kunnen complex zijn (doos met kruisje op dat opengedaan kan worden met daarin nieuw proces) → hi¨erarchisch gemodelleerd.

**Message flow:** Interactie tussen verschillende actoren.

**Gateways:** Beslissingen controleren, synchroniseren,...

**Slide 8:** BPM is taal, notatie, woordenlijst. Met alleen symbolen weet je niet hoe de taal in elkaar zit, dus nog een methode nodig. Kwestie van stijl: soms heb je om hetzelfde neer te schrijven 2 verschillende mogelijke diagrammen. In video: modelling alternatives.

**Slide 9:** Verschillende processen van abstract naar concreet naar uitvoerbaar geven aanleiding tot gebruik van verschillende symbolen, sets in BPMN:

1. Niveau 1: abstracte processen, globale plaatje. Basissymbolen, maar niet volledige pallet nodig.
2. Niveau 2: analytische BPMN: meer gedetailleerde symbolen.
3. Niveau 3: uitvoerbaar: ook kijken naar XML-bestanden die worden gegenereerd.

**Slide 11** : (Tool: academic.signavio.com → kan je zelf proberen.)

* Start event: cirkel met enkele, dunne lijn.
* End event: cirkel met enkele, dikke lijn.
* Gateway: beslissing. Geen symbool == X.

**Slide 12:** Elk process model staat model voor een reeks van processen die worden uitgevoerd, niet voor specifieke taken → process instance: 1 feitelijk proces dat we kunnen weergeven door middel van een token of meerdere tokens. Kleuren om de verschillende process instances van elkaar te kunnen onderscheiden. Start event cre¨eert tokens.

**Slide 13:** Ruiten geven weer waar een beslissing wordt genomen. Zowel om proces te splitsen in verschillende paden als om ze te laten samenkomen: split en join.

**Slide 14:** XOR Split: er zijn 2/meer alternatieve uitgaande paden aan de gateway: er komt er 1 binnen en er gaan er verschillende naar buiten. Er wordt maximaal 1 uitgaand pad gekozen, maar ook minimum 1. Die beslissing wordt genomen op basis van gegevens (data-based gateway). Schuine streep door ja: we gaan ervan uit dat als we de info niet kunnen vinden, dan is yes de default. Zonder tegenbericht is het dus de mogelijkheid met het streepje.

**Slide 15:** IOR Split: je kan 1/meerdere van de uitgaande pijlen volgen naargelang de conditie waar is. Je gaat ervan uit dat om de voorwaarde te checken, je data hebt (dus ook data-based) en afhankelijk daarvan ga je dan combinaties uitvoeren. Minstens 1, maximum 3 (hier). Gedraagt zich half als XOR en AND.

**Slide 16:** Parallel split: 2/meer uitgaande paden en al die paden worden gevolgd → token wordt gesplitst. In BPMN is het ook toegelaten om die gateway niet te gebruiken en gewoon 2 uitgaande pijlen te hebben want elke pijl zal gevolgd worden. In termen van stijl is het beter om altijd de gateway te tekenen.

**Slide 17:** XOR: de exclusieve OR zou je spontaan kunnen interpreteren als ”ik ga kiezen via welk van de 3 ik binnenkom en doe dan 1 keer de volgende taak” → NIET zo! Vanaf moment dat 1 van inkomende taken een token krijgt, wordt uitgaande pijl geactiveerd: alle tokens nemen die binnenkomen en 1 mag uitgevoerd worden nadat 1 van de voorgaande taken is uitgevoerd geweest. Er komt een token binnen via B, wordt doorgesluisd naar E, dan van C en dan van D. er wordt niet gesynchroniseerd, er worden geen tokens weggegooid, gaan allemaal door!

**Slide 18:** AND: gaat tokens samenvoegen. Wacht op token van B, C, en D. pas als ze alledrie in de gateway zitten, worden ze samengevoegd om ze dan verder te laten gaan. → Forceert synchronisatie.

**Slide 19:** IOR: gedraagt zich half zoals AND en OR: gaat zich voornamelijk gedragen als AND: gaat wachten op actieve paden. Stel dat je weet dat er via het onderste niks/meestal niks kan doorkomen: IOR gaat wachten op actieve paden. Kan dus achteruitkijken om te kijken of er tokens in aantocht zijn. → Iets intelligentere gateway. In vb: zal niet wachten op C want daar geen token in aankomst.

**Slide 21:** let op: zie Figuur 8.1.

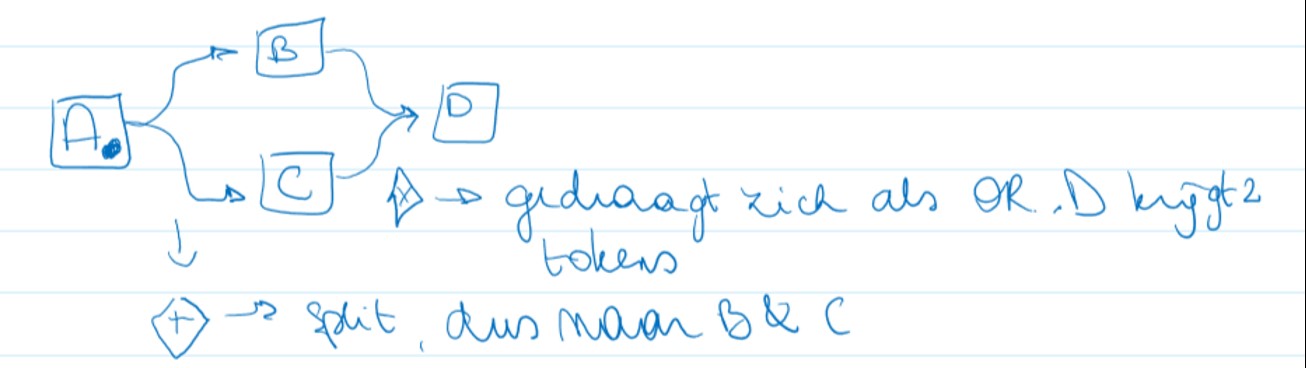


Figure 8.1: Duidende tekening bij Slide 21

Altijd gateway tekenen voor duidelijkheid!

**Slide 22:** Actoren voorstellen door middel van pools: draagt ofwel naam van proces of deelnemer in proces die het vertegenwoordigt. Belangrijk: proces kan nooit buiten pool gaan.

**Slide 23:** Pools kan je onderverdelen in lanes en die kan je onderverdelen in sublanes.

**Slide 24:** Sequence flow (pijl) kan nooit over pools gaan. Enkel via messages. Pools geven gescheiden partijen weer. Tussen de actoren worden berichten uitgewisseld en tussen de actoren worden boodschappen uitgewisseld. Binnenin een pool ga je geen messages gebruiken. Binnen een pool heb je alleen sequence flows.

**Slide 25:** Het lijkt alsof je 2 processen hebt die in parallel lopen, maar eigenlijk heb je een proces in 4 taken: plan shipping heeft input nodig en kan dus nooit starten voor request shipping is gestart. Volgorde waarin taken wordt afgewerkt is dus ook afhankelijk van verschillende pools. Taken kunnen wel op TODO-lijst komen te staan, maar moet soms dus wachten op message van andere pool.

**Slide 26:** Slecht voorbeeld: bovenaan klant die order gaat plaatsen: boodschap sturen naar supplier waar stock availabilty wordt gecheckt,... Geen goed vb: normaal heb jij als bedrijf geen inzicht in de stappen die de klant onderneemt. Je weet alleen welke info je naar de klant stuurt en wat je van de klant verwacht. We behandelen de klant als black box. Je tekent dus wel klantenpool, maar geen taken in die pool. Je weet alleen welke info ontvangen wordt van klant en welke gestuurd wordt. **Slide 27** bevat een realistische beschrijving.

**Slide 28:** Data: weergeven als individuele objecten of opslag (disk). De associatie kan een pijl hebben om aan te geven of input/output is. Als lezen & schrijven: undirected connection. Data store: data die wordt opgeslagen: ook ter beschikking als proces afgesloten is.

**Slide 29:** Dataobject is lokale variabele. Als proces ten einde is: info weg. Data store: data blijft bestaan na be¨eindigen proces.

**Slide 30:** Commentaar toevoegen aan proces. Examen: alleen pools, lane en control flow. Geen data-objecten!

**Chapter 9**

# Les 9: 09/03/2015

## 9.1 Slides: 3E BPEnactment

**Slide 3:** Procesdefinitie gebruiken om proces instances te initialiseren tokens genereren. Beheer van de tokens gebeurt door BPMS. Binnen PD: activiteiten: ook aantal activiteitinstances in het systeem. Bpmanagementsysteem gaat u helpen met het beantwoorden van vragen over het proces. Video: je maakt een procesmodel dat je kan uploaden in een systeem (BPMS), daarin heb je pools en lanes: taken toegewezen aan actoren. Zo kan je de voortgang van je proces gaan bewaken. Het systeem kijkt dat de volgende taak wordt gegeven aan de juiste persoon.

**Slide 5:** Databank met objecten (blauw). Daarboven draaien aantal informatiesystemen. Minder ideaal: elk IS heeft zijn eigen databank, geen gedeelde databank. Al die infosystemen hebben bovenop een business process management system (niet altijd → geen geautomatiseerde ondersteuning van processen of IS hebben dat deels ingebouwd).

**Slide 6:** Vorige slide is het ideale plaatje. Nu verschillende architecturen overlopen zoals ze voorkomen in de praktijk met verschillende graden van ondersteuning van bedrijfsprocessen: geen ondersteuning tot zeer intelligente en doorgedreven ondersteuning.

1 informatiesysteem en dat beheert zijn eigen gegevens (dus ook 1 databank, nl in IS) en geen geautomatiseerde proceslaag (daarom mens met gedachten → process in the brain). Die persoon denkt na wat er gebeurd is en wat er daarom moet gebeuren en wat daarvoor nodig is.

**Slide 7:** Meerdere IS’en georganiseerd volgens functionele domeinen. Typisch: end-to-endprocessen die doorheen verschillende functionele domeinen lopen. Bv onderzoeksproject komt binnen via FD ’onderzoek’, gaat via domein ’financi¨en’ nadat onderzoek is goedgekeurd, in domein ’HR’ vacature vrijgegeven,... 1 proces verbindt alle aspecten van het bedrijf met elkaar. Verschillende systemen en proces gaat daardoorheen.

**Slide 8:** Meer realistisch plaatje: verschillende IS’en met elk hun eigen data (dus niet ideaal!) en het proces zit in persoon zijn hoofd (moet dus onthouden wat nodig is voor elke stap).

**Slide 9:** Wat je wel ziet is dat men vaak zal proberen om enige vorm van ondersteuning te geven aan de mensen die in zo’n omgeving moeten werken. Als je losstaande IS’en hebt, heb je vaak dat dezelfde info in verschillende toepassingen nodig is. In vb op slide: van taak A naar taak D: info tussen 2 databanken wordt doorgesluisd. Men kijkt naar het proces en ziet hoe info van de ene naar de andere moet doorstromen om daarna de rode pijlen te programmeren zodat info van de ene toepassing naar de andere overgeheveld kan worden.

**Slide 10: Slide 9** toegepast op onze universiteit: KULoket en Toledo zijn 2 IS’en. Hebben beiden een eigen databank. Studenten worden verondersteld te weten wat te doen (dus process in the brain). Student maakt gebruik van KULoket om cursus te kiezen. Vanuit KULoket wordt er een inschrijving doorgestuurd naar Toledo (zodat je de info kan krijgen). Als je het vak leuk vindt (is keuzevak), geen probleem: in ISP laten. Anders ga je je uitschrijven in het ISP (in KULoket), maar wordt niet doorgegeven aan Toledo. Je moet dus jezelf manueel uitschrijven uit Toledo. Anders blijft dat vak staan in je lijst.

**Slide 11:** Voordelen automatiseren: rode pijlen: gebruiker moet geen info dupliceren: wordt automatisch doorgegeven door het systeem en wordt (correct) doorgevoerd. Nadeel:

* Je blijft met dubbele info/gegevens zitten, met alle gevaar voor inconsistentie (bv lijst van ingeschreven studenten op Toledo kan incorrect zijn wegens vorige slide. Op KULoket wel correct. → in juiste databank gaan zoeken!).
* Doorsturen van gegevens coderen: je procesorganisatie hardgecodeerd → maakt organisatie minder flexibel. Als je proces wil veranderen (bv in Toledo inschrijven en vloeit door naar KULoket in plaats van omgekeerd), moet je alles herprogrammeren. → Je gaat moeten wachten, maar je wilt je bedrijf flexibel en agile houden, dus als je je proces hertekent, wil je dat het IS zo kan volgen.
* Dingen die te maken hebben met werkorganisatie raken vervlochten met feitelijke functionele ondersteuning van applicatie.

**Slide 12:** Je moet een process engine gaan gebruiken en die gaat voor jou proberen om dat stukje procesbeheer te capteren en te ondersteunen. Op de slide heb je process engine met 4 stappen. PE gaat niet het werk doen dat geassocieerd is met die 4 stappen, gaat zorgen dat mensen/actoren weten dat ze een taak moeten uitvoeren. Functionaliteit blijft in het systeem zitten, maar proceslogica kan door process managementsysteem doen. Process engine gaat zien (zoals in video: aanvragen tot verlof) dat er iets is binnengekomen en weet dan welke stappen gevolgd moeten worden en stuurt de requests door naar de juiste actoren.

**Slide 13:** Ook in PE verschillende graden van ondersteuning. Meest eenvoudige: niet gekoppeld met IS. Kent processen en kan TODO-lijsten beheren en taken doorgeven aan volgende persoon, maar heeft geen link met IS. In min of meer ideale wereld: proces niet in de brain, is eigen infosysteem. Als mens ga je interageren met workflowsysteem en anderzijds gebruik je de nodige toepassing om je workflow te doen.

**Slide 14:** Als je verschillende IS’en hebt, ga je als gebruiker nog steeds rechtstreeks met workflowsysteem spreken en je hebt de user interface van de toepassingen. In het beste geval kan de workflow engine weten waar de nodige toepassing zit die de gebruiker nodig heeft (bv: weet dat voor taak X Toledo nodig is en voor Y KULoket).

**Slide 15:** Vbn: 2 workflows, supportmanieren met of zonder link naar UI.

**Slide 16:** Iets geavanceerder systeem: aparte takenlijst (in mailbox van vorige slide zit alles gewoon samen), niet meer vervlochten met emails, gewoon TODO list. Je krijgt een iets meer geavanceerde ondersteuning waarbij je veel meer info hebt over de taken die moeten gebeuren met start- & einddatum,...

**Slide 17:**

* Intelligentere ondersteuning betekent dat BPMS werk kan toewijzen aan actoren, kan dat overheen verschillende actoren doen. Kan werk routeren van de ene persoon naar de andere. In betere geval: load management ook doen (in het oog houden wie al veel werk heeft en wie minder, om zo werk door te sturen).
* Processysteem gaat taak direct laten uitvoeren in plaats van via tussenpersonen te doen. Of rechtstreeks in toepassing gaan en dingen klaarzetten zodat voor menselijke gebruiker makkelijker wordt: betere link met onderliggend IS.
* Faciliteiten voor activity management en performance evaluatie: kijken wat de stand van zaken van de processen is, hoeveel taken heb je, wie heeft hoeveel gedaan in hoeveel tijd? → Info en rapportering over hoe processen verlopen in de onderneming.

Bizagi video: geeft goed beeld wat erbij komt kijken met alles wat erbij hoort.

**Slide 18:** Bovenaan: BPMS met 3 mensen (met al dan niet dezelfde rol) waarover die het werk kan verdelen/routeren. Connectie gebeurt niet naar GUI, maar service interface. → Automatisch data ophalen uit het systeem, wegschrijven. Vereist dat IS webservice interface ter beschikking stellen.

**Slide 19:** Voordelen:

⊕ Multi-applicatiesysteem in meeste bedrijven.

⊕ Door BP Enactment: processen modelleren en opvolgen in praktijk: uitvoering proces begeleiden.

⊕ Co¨ordineren over toepassingen gebeurt door BP-engine (geen rode pijlen meer), in die sequentie zit data (gaat BPE zeggen), kan data uit IS1 halen en aan IS2 geven: data gaat nu via boven (zoals in **Slide 20**). → Co¨ordinatie niet ingebakken in applicatie → veel flexibeler.

⊕ Processen kunnen heel vlot aangepast worden.

Nadelen:

Je hebt nog altijd de mogelijkheid tot dubbele info (want elke toepassing heeft zijn eigen databank).

De replicatie en het doorsturen van gegevens gebeurt via BPME. Nadeel want maakt dat proces heel complex wordt omdat je tussen de echte taken door taken krijgt zoals ’request info out of IS1’. → Extra taken in procesmodel die meer te maken hebben met het feit dat het vanonder niet ge¨ıntegreerd is dan dat ze te maken hebben met de feitelijke werkorganisatie → complexer systeem.

PE is een toepassing. Als je er zo 1 hebt die alle werknemers van het bedrijf, kan dat voor een bottle neck zorgen. Als dat systeem crasht, ligt heel het bedrijf plat. Risico op problemen neemt toe naarmate dat zo’n systeem heel veel data moet verhuizen van de ene naar de andere.

**Slide 20:** Ideale wereld: 1 ge¨ıntegreerde databank. Soort van vision, target architecture, waar je naartoe moet evolueren binnen een bedrijf.

**Slide 21:** Beter, bovenop voordelen van daarnet (lichtblauw):

⊕ Geen replicatie van gegevens, maar 1 registratie van gegevens.

⊕ Co¨ordinatie op vlak van data tussen verschillende toepassingen gebeurt door middel van de databank. Als klant van adres verandert, weten alle toepassingen wat het nieuwe adres is van de klant. Anders zou het kunnen dat marketing het bv het wel weet, maar facturatie niet. Nu: 1 unieke waarheid en 1 unieke registratie van gegevens.

⊕ Processen worden een pak eenvoudiger. Er is veel minder gegevensoverdracht tussen verschillende taken. Doorgeven moet niet meer gebeuren op niveau van procesmodel, zit in databank. Processen worden eenvoudiger en zuiverder.

Nadeel:

Wanneer je toepassingen gebruikt die niet kunnen communiceren met centrale databank. Als je meerdere databanken nodig hebt, blijf je risico van duplicatie hebben en extra complexiteit.

**Slide 22:** Daarnet: een van de redenen waarom we met PE willen werken en dus niet co¨ordinatie tussen toepassingen willen hardcoden is dat de bedrijven flexibel moeten zijn, processen moeten permanent aangepast kunnen worden. Processen kunnen aanpassen is weer probleem op zich. Hoe ga je procesevolutie ondersteunen? 2 grote manieren waarop je dat kan ondersteunen. Nieuwe versie van proces implementeren, maar aan het as-is model hangen instances.

1. Simple versioning: processen die al runnen laat je zoals ze zijn: 2 versies van het proces in je systeem. De oude versie blijft in je systeem lopen tot alle oude process instances zijn afgehandeld.
2. Evolutionary change: instances die opgezet zijn in oude versie overzetten naar de nieuwe.

Voorbeeld van pensioenen: mensen die met brugpensioen gingen willen simple versioning behouden, regering wil evolutionary change.

**Slide 23:** Voorbeeld van proces: is het oude proces dat gaat over schadevergoeding bij ongeval.

**Slide 24:** Veranderen: keuze: in schema 1 had je al 3 PI’s, die migreer je naar 4,5,6 en dan voer je weer hervormingen uit en krijg je 7,8,9. Elke instance blijft in zijn versie draaien. Bij evolutionary change: migreren.

**Slide 25:** De process engine, veel daarvan zijn gebouwd toen BPMN nog niet bestond, dus veel PE’s hebben interne logica. Wat jij tekent in BPMN kan dus anders runnen.

**Slide 26:** Process enactment afgerond. We hebben ons proces ge¨ımplementeerd, zo komen we tot process monitoring and controlling. We hebben heel veel info over wie wat wanneer doet, in welke toestand processen zich bevinden,... → Heel dashboard van key performance indicators. ⇒ Process discovery: je gaat geen processen ontdekken door met mensen te praten, maar door gewoon te gaan kijken wat er echt gebeurt, in de praktijk: process mining → terug tot inzichten komen die ervoor zorgen dat processen herontworpen worden.

**Slide 28:** Kwaliteit wordt slechter doorheen de tijd. Mensen krijgen een procedure opgelegd, maar na verloop van tijd sluit die procedure slechter aan op de realiteit. Je MOET veranderen, constant evolueren met de omgeving. Proceskwaliteit goed houden: permanent aandacht aan besteden en geleidelijk aan verbeteren. Of periodisch aandacht aan besteden, of beiden doen (zie verschillende pijlen op slide).

**Slide 29:** Voorbeelden van dashboards onderaan: BAM. Je kan daar vanalles mee doen: bij orders gaan kijken wat de doorlooptijd is van orders, hoeveel worden succesvol geleverd, hoeveel geannuleerd,... → De dashboards zijn drivers voor BPI, zowel op continue basis, als wanneer je bijzondere aandacht wilt besteden.

**Slide 30:** Typische zaken waar je naar gaat kijken:

* Volumes, bv in unief moeten financi¨ele transacties worden goedgekeurd. Mensen met doorgedreven opleiding van hoe je iets correct ingeeft mogen de boekhouding doen. Je gaat kijken naar welke mensen dit snel en correct kunnen doen. Volume van studentenvragen: kijken of jaar na jaar stijgt/daalt, waarover vragen gaan,...
* Snelheden: wat is de gemiddelde snelheid waarmee een vraag in de onderwijshelpdesk wordt beantwoord? Je kan de snelheden gaan onderzoeken, zeker als je daarbij customer satisfaction neemt: wanneer worden klanten ontevreden? Als grote wachttijden: zien wat je eraan kan doen, wat is bron van de wachttijd.
* Errors: orders die geannuleerd worden, betalingen die niet correct gebeuren. Alles wat fouten genereert vraagt veel correctietijd → kosten drukken: fouten zo klein mogelijk houden. Onderzoeken op exceptions.
* Special cases: meestal bij processen het happy path (pad dat meeste PI’s gaan volgen want normale route), maar ook bijzondere route voor bijzondere gevallen. Bv topsporters, studenten met bijzondere faciliteiten,...

**Slide 31:** Eens je inzicht hebt: input voor process redesign: op basis van statistieken, maar ook model analyseren. ⇒ Input voor PR en dan weer evolutionary change of simple versioning.

**Slide 32:** BP-Improvement vs BPRE-engineering BPI: zoekt naar graduele verbeteringen, klein, lokaal.

BPRE: gaat vanaf nieuw blad opnieuw nadenken over hoe we innovatief kunnen zijn en totaal nieuwe manier van werken verzinnen.

Theorie¨en: Lean (lenig en slank: gaat zich concentreren op verwijderen van afval. Gaat zoeken naar nutteloze zaken en op die manier proces doen afslanken).

**Slide 33:** Geen radicale wijzigingen. Heel kleine verbeteringen defini¨eren op korte tijd. Bv: verandering formuleren: in welke richting, maat opgeven, proces opgeven, oplossing en doel.

**Slide 34:** Clean slate: vanaf 0 vertrekken, alles weggooien. Heeft meestal tot doel om de klant te pleasen: customer orientation, customer value zo hoog mogelijk. Zij gaan uit van het feit dat kleine optimisaties niet nuttig zijn (niet altijd terecht). Kwestie van change management: nieuwe processen invoeren: mensen meekrijgen die de processen moeten uitvoeren.

**Slide 35:** Vb van lean-theorie: zoek naar processen die waarde cre¨eren voor de klant. Je hebt dan alle stappen van het proces. Kijk voor elke stap of die waarde toevoegt (indien nee: afval, proberen te verwijderen). Je gaat je flow misschien moeten herconnecteren. Werk alleen als het nodig is. Begin niet preventief/proactief vanalles te doen, alleen werken als er echt een vraag is. Blijf jezelf in vraag stellen.

**Slide 36:** Drank in flessen, verpakt per 6, gestockeerd waar ze verpakt geweest zijn, naar stockplaats gebracht, daar op vrachtwagen geladen en naar klant gevoerd. Je gaat je bij elke stap afvragen waar de waarde zit. Feit dat drank in winkel is geraakt, is toegevoegde waarde. Drank in fles is ook toegevoegde waarde. Feit dat flessen per 6 worden verpakt: ook toegevoegde waarde want pakken van 6 zijn makkelijker verpakt (en misschien verkoop je zo meer). De flessen stockeren op paletten: vereenvoudigt vervoer → toegevoegde waarde. Paletten verhuizen: geen toegevoegde waarde: voor klant maakt het niet uit! Paletten in vrachtwagen laden heeft wel toegevoegde waarde. Dus 1 stap waarbij paletten worden verplaatst, heeft geen toegevoegde waarde voor de klant. ⇒ Laadstation → verhuizen naar waar pakken van 6 op paletten worden gestockeerd.

**Chapter 10**

# BPMN AM Activities v Les 10

## 10.1 Slides: 3D BPMN Advanced

### 10.1.1 Video: BPMN AdvancedModeling: Activities

**Slide 3:** First we look at when activities exactly start and end.

First: 2 activities in sequence. As soon as the first activity is finished, task 2 will start automatically and immediately. When task 2 is an automated service, it means that it will be automatically and immediately executed. When the SP is a human, then the task becomes available on the work list of that person. Might be notified to the user that the task is available. Means that the task may stay for a while without being executed. Even though the task stays in the TODO list for a while, we assume that the task has started. So if you want to keep track of how long the task is being executed, the waiting time (when nothing is really performed) is also counted.

**Slide 4:**

* In terms of completing tasks, the thing that completes the task is the thing that triggers the flow out of an activity. The next task/gateway will be started. A subprocess is completed when all its parallel paths have reached an end.
* In terms of abnormal completion: task can be interrupted by an activity, subprocess can be interrupted by various exceptional conditions. You don’t know what the internal state of an activity is, so you don’t know at what stage the activity is. You will use an event attached to a boundary to mark an abnormal activity. If a process doesn’t finish in time, the exception flow is followed (with the clock), otherwise the other arrow is followed.

Keep in mind that events and gateways have no performers. They are control logic elements, they are instantaneous, start immediately and complete immediately and have no duration.

**Slide 5:**

* Simple task: you can give a certain label according to the nature: sending/receiving messages. Human/automated performers.
  + Service task: task performed by application (automated).
  + User task: performed by human being that will use an application or make use of the business process engine.
  + Manual task: no application support, entirely manual.
  + Script: by business process engine: automated.
  + Business rule task: rules are checked by a rule engine.
* Complex task: cross at the bottom: box that you can open, inside the box you will find a new process.

**Slide 6:** BPMN subprocess: they are something that you can use if you want to visualise the end-to-end process (if whole process is too big to fit on 1 page). Allows you to do top-down modelling. Coarse-grained at the top and more fine-grained defined processes at bottom of hierarchy. Also for boundaries and scope.

**Slide 7:** 2 examples of subprocesses.

* Top: 2 activities: *take a break* consists of *read book* and *work in garden*. Subprocess might have explicit start and end, this isn’t true here. If *take a break* is activated, both subprocesses start simultaneously. It ends when they are both finished.
* Bottom: subprocess has received tildemark: means ad-hoc subprocess: every activity that’s embedded in the subprocess can start and can start any number of times. *Read book* could mean that you can read more than one book. There is not a fixed number of times each activity can be done.

**Slide 8:** CALL activity: used when you want to make your subprocesses reusable or define it as global task/subprocess that is available to all the processes → thick border. Seen as independent subprocess, in contrast with subprocesses embedded in and only known in encompassing super-process. Example: left one will only be known into the super-process, the one on the right can be used in any process because it is a call activity, can use it from any type of process.

**Slide 9:** Event: processes that can be started at any point in a global process → not part of normal flow of its parent process, not connected to this general flow of the parent process, there are no incoming or outgoing sequence flows. They are put as an independent box into the pool. They have a trigger, each time the start event is triggered while the parent process is active, the event subprocess will start. This may not occur or multiple times. You can say whether the process is interrupting or non-interrupting.

**Slide 10:** Example: global process = daily routine. 2 event subprocesses, not connected to global process, independent boxes put in the pool of the global parent process. First event subprocess has event phone rings: interrupting start event because full line. When the phone rings, the global process is interrupted and the phone is answered. When answer phone has ended, the global process is terminated as well because it has been interrupted. Second subprocess: noninterrupting (dashed line): means that during the 3 tasks of global process, the event can happen while other tasks are happening. When the task is finished, the global task will continue. Interrupting event subprocess may not happen or may happen once and non-interrupting subprocess may not happen or happen multiple times.

**Slide 11:** Activities have to be repeated: loop activity: equivalent of do-while statement. It means that you repeat it sequentially and you always check a condition to see if you’re going to continue the activity. Multi-instance: forloop: you know in advance how many times the process is going to be repeated. Vertical lines: in parallel, horizontal: sequential. Loop activity: sequential.

**Slide 12:** Example shown with parallel split.

### 10.1.2 Video: BPMN Advanced Modeling: Events

**Slide 14:** Events are indicated with a circle and are used to indicate that something has happened. They can affect the process, start it and end it. There can also be something in between like pause and resume. Received and sent events. 3 types: dependent on where they happen in the process.

1. Start event: indicates where the process starts.
2. End event: where the process ends.
3. Intermediate: in between start and end.

Many different types of events because the basic symbols can be adjourned with other symbols.

**Slide 15:** Overview of possible events.

* Start events: single thin line when interrupting, thin dashed line when non-interrupting.
* Intermediate events: double thin lines when interrupting, double dashed lines when non-interrupting.

White envelope: receive message. Black: send. Start always receives, end always sends, intermediate can both receive and send.

**Slide 17:** Start event: thin circle indicates where the process starts. Usually only 1 start event. It is allowed to have multiple start events. When it’s noninterruptive, the start event is drawn with a single, dashed line. Inside the start event you can indicate what the trigger for that start event is. When it is a blank, then it is any trigger, unspecified. If it’s a message: envelope: message directed to this process. Timer: scheduled process (e.g. every day at 9 o’clock). Conditional: watching data for some condition to become true. Any signal that is broadcast to any listening process. Hexagon: any of multiple signals. Plus: you wait for occurrence of multiple events at once.

**Slide 18:** One start event: good practice. It’s also allowed to have no start event. That way, any activity that has no incoming arrow is enabled and can start. Example of subprocess with 2 subprocesses: no start events for them. All the activities that have no active incoming flow have an implicit start event and an implicit parallel gateway. In certain cases you can use multiple start events: channel-dependent start.

**Slide 19:** End event: thick circle: indicates where process ends. No end event: implicit termination of activities with outgoing flow. Multiple end events possible. Good practice to use more than one for each distinct end state, but also to use a maximum of 2 end events (e.g.: one end event for normal ending, one for abnormal). Each path must reach an end for the (sub-)process to complete normally, so when you have multiple end events, all these parallel paths must be finished before the process ends in its entirety.

There is one exception: termination, error, abnormal process: abort the process, even if parallel paths are not yet completed. End event may specify a result signal: throw a certain signal. Envelope: throwing message: when process ends, a message is produced and sent to an external participant. Triangle: used for signal: broadcast signal, not addressed to particular process, all other processes can listen.

**Slide 20:** Terminate event example: process starts with technical review and financial review. If the finances are not okay, the process ends, even though the technical review may still be in consideration. If the finances are okay, wait for the technical review to end.

**Slide 21:** Intermediate events: allow you to pause and continue the processes. Semantics depend on border style and how it has been placed in the diagram. Placement can be in the sequence flow (arrow going in/out of event) or attached to activity: placed at boundary of event.

Full line: interrupting, dashed: non-interrupting.

Filled symbol: generate the event. Empty: catch the event. All the boundary events are of the catching type.

**Slide 22:** Example: start event: intermediate event with black envelope: throwing event: generate a message and send it out to process 3. Throwing an event is possible for only some of the types (you cannot throw time). The catching event type is where the white envelope is: you’re waiting for the event to happen, you have no control over when it will happen (e.g. with time). It’s not possible for an error in the flow to be caught: it will be thrown.

**Slide 23:** Event attached to an activity: *pay* and event is with a full line: will interrupt. How do you determine duration of the timer? When *pay* activity starts, the timer starts too (at same time). The activity *pay* starts and timer clock starts and if 2 hours have gone by and the *pay* activity is not completed, it will be interrupted and the bank will be contacted. Otherwise you will follow the normal flow and go to the end event immediately.

**Slide 24:** Example of non-interrupting intermediate event, we will not interrupt the activity, follow the exception flow, but you do follow the normal flow. You continue the activity of waiting while you contact your husband. If the one hour had had a full line, you would stop waiting, notify your husband, but not subscribe your child to the sports camp.

**Slide 25:** Event needs to have a certain scope. In the previous example: attached to activity. If the timer can occur across more than 1 activity, you will use a subprocess to delineate the scope of the interrupting event. The main path of the process is the upper one. At a certain point during the subprocess, it will be 8 PM so I have to send the kids to bed. It can go off during either event. If I finish relaxing before 8 PM, you might never send your children to bed. The 8 PM timer can only go off while the subprocess is running.

**Slide 26:** Use the intermediate events to wait for the first events that will happen. Sometimes you have 2 possible events that can happen, you will wait for the first that happens and ignore the other one. You need an OR-gateway: event-based-OR-gateway (in contrast to data based gateway): diamond with inside it a multiple intermediate event. In the example: send request: either receive response or 2 weeks pass. Either one of the right actions will be chosen. Event-based. Not based on data, but on the first event that happens. You have no control over what is going to happen.

**Slide 27:** Throwing, catching and signalling of events works between different parts of processes. If you work with messages, it only works between 2 pools. You cannot throw-catch the same message within the same pool. Error and cancel throw-catch doesn’t work between pools, but between a subprocess and the boundary of that same subprocess. The error and cancel will be end-events and you catch them as intermediate events at the boundary of that subprocess. They can never be caught as a start event or an event-based gateway.

Escalation throw-catch also works with a subprocess, but it can be thrown as an intermediate event or as an end event and again to the boundary of that same subprocess.

**Slide 28:**

* Error-event: example: subprocess: check payment: either normal end: proceed, or abnormal event: throw error: caught as intermediate event attached to the boundary of the subprocess.
* Escalation: also allowed to be non-interrupting.
* Cancel: similar to error, but main difference is that cancel is only used to end transactions.

**Slide 29:** Signal throw-catch can happen within different lanes of a same process. Negotiation with client is a subprocess, can have an error that is caught at the boundary of that subprocess and then you broadcast that the negotiation failed.

### 10.1.3 Video: BPMN Advanced Modeling: Modeling Alternatives

**Slide 31:** Example: process to refund expenses: express process logic with the 3 given events/subprocesses. After that similar constructs.

**Slide 32:** Reimbursement of expenses: process information: person has expense report and files this. Check if this person has an account. If not: create a new account. Then the report is reviewed for automatic approval. Might require approval of supervisor, if it is rejected: notice is sent. Reimbursement goes to the employee’s direct deposit account. Other rules: check slide.

**Slide 33:** Basic process with first attempt to model the desired behaviour with events. Happy path == upper path, otherwise, the path with the timer is followed. Everywhere a terminate event instead of normal end event. Suppose this process goes very fast and ends here, normally, it terminates the parallel path as well → approval in progress e-mail is not sent and cancellation isn’t sent either. If branch takes 10 days: after 7 days: approval in progress. After 10 days reviewed: lower branch is cancelled and cancellation notice is never sent. If the path takes more than 30 days: after 7 days: in progress, after another 23 days: cancellation notice and abort. ⇒ It nicely works and it does what we expect it to do.

**Slide 34:** With boundary events: put normal flow in subprocess. Captured in a box so that we can use it as scope delimitation for 2 events (interrupting & non-interrupting event). You obtain the same event, the same logic, but a better distinction between normal flow and the interruptions, things that happen according to certain timings on top of the normal flow.

**Slide 35:** With event-subprocesses: again: normal flow which has normal end events. 2 subprocesses which will be triggered after 7/30 days.

**Slide 36:** Decisions & rules: similar but different. Processes can contain a lot of rules and you can have a more declarative approach to business processes where you only formulate rules. To a certain extent, rules and processes are complimentary, they each allow you to model processes from a different perspective and can complement each other. No guidelines on how to combine the two together.

Ideally, business rules should be defined independently of the process in which they are used. The rules have to be general & observed in different processes. If you change a rule, you would like to see the change in all of the processes where these rules are applied, they should change too, or change accordingly.

This would be a reason to define the rules outside of business processes. Different places where you can put rules:

* Gateways: routing rules: process-specific! → Not a good place to put a rule here that is process-independent. If it were to change, you’d have to change it in all the different places.
* Business rule task: process-independent. Task that is a decision service that you can call at a particular point in the process. Allows you to call upon a service that is outside of the process and can be checked. Task can return true/false. You can put a gateway to test the outcome of the decision process and then route the process in the right direction.
* Conditional event: can be used to signal when a particular condition has become true: observing certain data values and having it test on these data values.

**Slide 37:** Sending a message == task: wait for a message: receive task/catching message intermediate event. If you wait for a message, it’s not the same as catching a message on the bounds of a subprocess, because in the latter case, you’re not actively waiting, the message might arrive when you’re doing something. When the message arrives at another time, you don’t let yourself be interrupted by that message. When you’re waiting for a message and want to model the time-out: gateway: XOR: choice between waiting for a message or a until certain deadline has been reached and you proceed with something else.

**Slide 38:** Sending tasks: send task == simple task that is designed to send a message to an external participant relative to the process. Once the message has been sent, the task is completed. If you look at the definition, it’s the same as an intermediate send message event. In both cases: communication between process and outside entity (same for both), difference: event has no duration, task does. When the composing of the message is simple (doesn’t take time): event. If the message-composition takes a considerable amount of time, a task is more appropriate. If you have the possibility of message communication, but not the certainty, then you have to use a user task.

General advice: user task for human communication ve-mail/phoning. Not sure that there is a message that will be sent: send message throwing event for machine to machine communication, especially if it regards very simple messages.

**Slide 39:** Don’t use message task/message event for communication within a process. Communication between lanes: sequence arrows. Not necessary to have a send task to symbolize that work is passed to the next person, this happens automatically and is part of the functionality of a business process engine. Internal communication with a person might not be actively in the process. If that person does not have a work list: simply use a task like ”notify manager”.

**Slide 40:**

• Service task: automated task executed by something other than process engine, typically: processed by application. Application will be invoked by process engine.

Script task: executed on the process engine itself. If you want a synchronous call to a service, the process will wait until the task is completed and it sends a reply → not for long-running processes.

Asynchronous service request: launch outside of application, but don’t wait for it to be finished. You model the service as a black box pool and you send a message to it, to indicate that it can start.

**Slide 41:** Process 1: send message to SP. At a certain point in time you receive a response/error → process it. Between this and the gateway, you can put a number of tasks. Normal task and a service task where you don’t model the communication with the application, you just wait until it finishes and then proceed. Service requests: waiting until short running service is finished to request a long running service. Both could be modelled by means of intermediate events.

**Chapter 11**

# Huistaak: BPMN Quiz

## 11.1 Slides: BPMN Quiz

**Vraag 1** Antwoord: C is correct.

* A is fout omdat je de taak gaat doen en nadat je die hebt gedaan, ga je in elk geval door naar fulfill order. Je doet niks met de beslissing die binnenin is gebeurd.
* Hetzelfde geldt voor B. De terminate event be¨eindigt alleen dat niveau van het proces. Be¨eindigt alleen het parallel pad daarbinnen. Terminate kan alleen op bepaald niveau iets be¨eindigen. Terminate event heeft dus geen effect op het totale process.
* In D, gebruik van die error event is fout: als je een error throwt, zou je die moeten vangen. Dus als je een error genereert en er niets mee doet, is het onjuist gebruik.

**Vraag 2** Antwoord: B is correct. Verschil tussen lus en multi-instance taken: multi-instance taken die met horizontale/verticale streepjes zijn: bij aanvang van die taak weet je hoeveel keer je die gaat uitvoeren. Bij een loop is het niet gekend! Receive applications: je weet niet hoeveel je er gaat ontvangen en je ontvangt ze ´e´en voor ´e´en. Eens de sollicitaties binnen zijn, weet je hoeveel interviews je gaat doen, dus je mag dan wel streepjes doen. Plusje: complexe taak: interview op zich loopt niet per s´e in parallel. Het hele proces kan je in parallel opstarten voor alle kandidaten. Wil je het sequentieel doen: eerst kandidaat 1 opbellen, alles afhandelen, dan kandidaat 2,...

**Vraag 3** Antwoord:

* A: ik ga documenten vragen. Het opvragen van die documenten zou langer dan 2 weken kunnen duren, dus het stellen van de vraag wordt onderbroken na 2 weken (fout dus!).
* B: documenten opvragen en als die taak gedaan is, wacht ik 2 weken. Na die 2 weken ga ik 2 dingen doen: rejection sturen en claim afhandelen. Je zegt dat je ze niet gaat afhandelen en gaat ze dan wel afhandelen.

C: stuur de vraag om meer info. Taak *receive documents*: als die langer duurt dan 2 weken, ga ik een rejection sturen. Is die receive docs klaar in minder dan 2 weken tijd, ga ik de klacht verder afhandelen → **Correct**.

* D: ik doe een taak en als de hele taak meer dan 2 weken duurt: stuur rejection. Als de 3 taken samen langer duren dan 2 weken, onderbreek ik dat (kan gebeuren tijdens request, receive en process claim).

**Vraag 5** Antwoord:

* A: event-based gateway: proces gaat moeten kiezen tussen alternatieve paden. Maar keuze ligt niet in uw handen als procesuitvoerder. Pad dat gaat gekozen worden, is event-based. Je gaat een antwoord ontvangen, of er gaat een week voorbij zonder antwoord te ontvangen. Beginnen met event-based gateway is dus juist. Antwoord kan positief of negatief zijn. Bij approval: order uitvoeren en confirmation sturen.
* B: begint met data-based gateway, beslissing moet gebaseerd zijn op gegeven die ergens opgeslagen of geschreven zijn. Er staat nergens of er een antwoord gaat komen. Je hebt nergens info. **B is dus fout.**
* C: is een variante op A: ”ik ga de taak ”wachten op antwoord” doen” en ofwel is die taak binnen de week voorbij, dan kijk ik naar antwoord en handel ik accordingly. Binnen week geen antwoord: send rejection.
* D: zegt juist hetzelfde als A. In plaats van de enveloppe open te doen, weet je meteen wat het antwoord is. Maar voor de rest dus exact hetzelfde.

**Vraag 6** Antwoord:

* **A is juist**: ik start het proces met een taak en in die taak heb ik A en B en daarop zet ik een timer. Dat betekent dat de klok van 1 uur begint te lopen vanaf dat het binnenste startevent is opgetreden. Die klok begint effectief te lopen vanaf dat het proces in zijn geheel gestart is. Gaat kijken hoe lang de combinatie van A en B duurt. Indien meer dan 1 uur: stopgezet & naar C. Anders: be¨eindig proces.
* B: begin proces met 1 uur (om 1 uur of na 1 uur, niet eens duidelijk). Je doet A, hoe lang dat ook mag duren, dan B, hoe lang dat ook mag duren, signaleer deadline en doe dan C. Alles mag zo lang duren als het wil.
* C: begin met start error event (kan zelfs niet!). Je doet A en B zolang als nodig, dan wacht je een uur en dan doe je C zolang je wilt.
* D: je start met een klok die een deadline heeft (whatever that means), je doet A zolang als nodig, dan doe je B en als B op zich meer dan een uur duurt, ga je naar C. Dat is niet wat er gevraagd werd: er werd gevraagd within 1 hour of *process* start, not *task* start.

**Vraag 7** Antwoord:

* A: event-based gateway, maar dan moet je events hebben na elke tak. Fulfill order is geen gebeurtenis waarop je wacht, is een taak die je moet doen.
* B: gateway gevolgd door event receive cancel. Kan je niet beslissen op basis van gegevens of dat gaat gebeuren of niet.
* **C is juist**: order klaarmaken voor verzending en terwijl je daarmee bezig bent, kan je onderbroken worden door een bericht dat het order wordt geannuleerd. Als je onderbroken wordt, ga je dat bevestigen. Was je al klaar, ga je het alsnog versturen (klant was te laat met cancellation aan te vragen).
* D: 2 dingen in parallel doen: fulfill order en tegelijk wachten op receive cancel. Pijl met kruisje cancel en interrupt cancel, bedoeld als throw-catch (zwart kruis throwt en wit kruis catcht). Cancel is enkel van toepassing bij transacties. → Fout gebruik van throw-catch.

**Vraag 9** Antwoord:

* A: inpakken wordt niet onderbroken, ship method wordt geupdated en je gaat hier ook nog eens naar verschepen. Token gaat sowieso door naar ship order. Indien de non-interrupting event optreedt, zal de token uit fulfill order niet weggaan, gaat een eigen token maken die naar update ship method gaat (dus 2 tokens in systeem). Gaan dus alletwee een order shippen. 1 keer volgens de oorspronkelijke levermethode, ´e´en keer volgens de gewijzigde methode.
* B: fulfill order heeft een token. In normale omstandigheden: ship order. Als update ship message: fulfill order wordt onderbroken. Als inpakken nog maar halfweg is, pech, je gaat shipment method aanpassen en zo leveren. Ook niet ok.
* C: C zegt fulfill order, daar zit een token en als dat klaar is, gaat het token door naar de gateway. Als er een ship method update komt, gaat het het oorspronkelijk token ongemoeid laten en zijn eigen maken. Als dat klaar is, gaat het ook door naar de gateway. XOR gateway: is pass-through, gaat geen keuze maken tussen 2 tokens die binnenkomen. Volgende taak mag uitgevoerd worden als fulfill order is gebeurd, maar ook als update ship method is gebeurd → 2 keer geleverd. A en C zijn eigenlijk exact hetzelfde schema.
* **D is juist**: inclusive OR: je kan de logica zetten van ’als het binnenkomt via fulfill order alleen, zie ik dat er tokens binnenkomen via beide routes, dan ga ik een keuze maken en wachten tot zei beiden zijn aangekomen en ga ik maar 1 keer verder’ → maar 1 shipment en die gaat volledig zijn.

**Vraag 8** Antwoord: notify the customer, maar request wel nog afwerken! **B is correct.**

* A: handle request onderbreken, is niet de bedoeling.

B: handle request kan zo lang duren als je wilt, maar na 1 uur notify je de customer.

* D: nog een uur wachten tot je de customer gaat notifyen.

**Vraag 10** Antwoord: je begint met 2 dingen tegelijk, als klaar: begin met C. als A langer duurt dan een uur, stop met B en ga verder met C. **C is correct.**

* A: A wordt onderbroken, is niet de bedoeling.
* B, C, D: non-interrupting timer van 1 uur → goed.
* B: terminate-event: be¨eindigt alles en ook taak C gaat niet uitgevoerd worden.
* D: throw-catch met error-event: mag alleen van subtaak naar de rand van die subtaak: van binnen de doos naar de buitenkant van de doos. Niet tussen alternatieve paden binnen 1 proces, gebruik daarvoor signal-event zoals in C.
* C: throw-catch met signal event.

**Chapter 12**

# Les 11: 16/03/2015

## 12.1 Slides: 4ABSummary

**Slide 1:** 1 laag: scope, andere: ER-model, derde laag: hoe concreet ondersteunen met behulp van IS’en? Aantal kolommen: *Hoe?* Komt ook data aan bod: wat hebben we nodig? Je kan die heel taak-specifiek bekijken (wat heb ik nodig voor die taak), maar ook heel algemeen naar waarover gaat de organisatie structureel? → businessobjecten ⇒ kolom *Wat?*

Beter om het generiek te houden en niet taak-specifiek te doen en een generiek datamodel te maken!

**Slide 2:** Eerste laag: SCOPE: afbakenen welke informatie de organisatie nodig heeft. Probeer volledig te zijn: zowel de bedrijfsprocessen van vandaag als van morgen worden ondersteund door de data die worden bijgehouden en verzameld in de onderneming. Er is ook een bepaalde kost verbonden aan het maken van een datamodel en het implementeren, dus probeer ook economisch te werk te gaan en er dus niet teveel mee te doen. → Volledigheid en spaarzaamheid.

**Slide 3:** Gegevens modelleren met datamodel. Kijken naar businessconcepten of businessobjecten. We gaan kijken ’wat is een persoon, wat is een vak?’. Ga je kijken naar het vak in het algemeen, of kijk je naar de variaties per jaar? De evaluatievorm kan bv elk jaar veranderen. Maar misschien vindt men dat de doelstellingen van het vak vrij constant moeten blijven. → Misschien wil men een onderscheid maken, of net alles jaarlijks wijzigen. Je moet overeenstemming daarover bereiken als je alle informatie gaat opslaan. Als je het onderscheid maakt, moet je weten dat de titel en de doelstellingen, of die horen bij het academiejaaronafhankelijke vak of net bij het academiejaarvariante vak? Je moet dus gaan kiezen wat je waar zet. Als je het onderscheid niet kent, per jaar en daarvan weet je dan wat de titel is, doelstellingen,... → Ga kijken hoe het bedrijf werkt. Als een student is ingeschreven voor een vak: academiejaaronafhankelijk of net niet? Allemaal uitschrijven en in schematechniek neerschrijven (zoals in BPMN), bv ER (Entity Relationship), EER (extended ER) en UML.

**Slide 4:** Gedaan met enterprisemodel naar information layer. Hoe wordt het ER-model omgezet naar een database schema? Information needs is iets dat

wordt gedaan door eindgebruikers (wanneer ze data nodig hebben, kunnen ze terugvallen op pregefabriceerde rapporten (bv bankafschrift) of ad hoc vragen (geen van de voorgedefinieerde rapporten voldoet aan hun noden, ze krijgen dan de mogelijkheid om zelf de databankte ondervragen met behulp van SQL)). We gaan kijken hoe de opslag in elkaar zit. Je hebt veel zaken die te maken hebben met de kwaliteit van hetIS, maar wordt niet behandeld in onze cursus.

**Slide 6:** Boek staat misschien ongeveer in hoofdstuk 3 van de cursustekst.

**Slide 7:**

* Mensen ondervragen: requirements collection and analysis, nadenken over toekomstige scenario’s en analyseren.
* Data model daarvan maken. Ofwel via ER ofwel via EER.
* Relational model: voor databank: mapping van het ene naar het andere

(laatste bolletje).

**Slide 8:** Wat houdt het in? aankoop- en leversysteem: een aantal processen (aankopen, levering,...). Voor zo 1 proces kan je gaan kijken welke activiteiten daarin zitten en welke dataelementen nodig zijn. Procesanalyse blijft een belangrijke bron. Daarvoor heb je dan information requirements.

**Slide 9:** Wie doet dat soort werk? Vrij moeilijk want je praat met verschillende mensen die niet allemaal hetzelfde vertellen. Heel wat analysewerk dat erbij komt kijken. → Werk van een informatieanalyst (informatiearchitect). Niet per s´e een IT’er, maar iemand die goed kan zien wat de gebruikers willen, goed kan communiceren en dan goed kan capteren wat er dan nodig is. Heel veel domeinkennis nodig, iemand die in een bepaald domein werkt als informatieanalyst, daarna zijn IT-gerelateerde zaken laat varen en wordt beschouwd als expert.

**Slide 10:** Je capteert je datamodel en je probeert daarmee de werking van de organisatie zo goed mogelijk te vatten en neer te schrijven. Later wordt dat ER-model vertaald naar een relationeel model.

**Slide 11:** Hoe onstaan: Peter Chen, is uitgegaan van databanken (verschillende soorten) en hij wou een technologisch onafhankelijke manier om op te schrijven wat nodig was. → ER-model. Is expressief en geeft een goede samenvatting. Componenten: entiteittypes, attribuuttypes en relatietypes.

**Slide 12:** Entiteittype: bv supplier, product, purchase order. entiteit*type* want vorm van classificatie: we spreken niet over leveranciers in het algemeen, maar over *een* leverancier. Relaties worden weergegeven met een ruit met daarin altijd 2 namen. Attributen in ellips.

**Slide 13:** Entiteit is een object in de re¨ele wereld (ik, lokaal,...) → gegroepeerd tot entiteittypes. Kunnen fysiek zijn, maar het vak zelf is ook een entiteit (ook al is het niet tastbaar). Elke entiteit heeft eigenschappen, attributen. Elke entiteit kan je karakteriseren aan de hand van attributen die een bepaalde waarde hebben.

**Slide 14:** Je moet de wereld zien als (een) verzameling(en) van objecten. Hele theorie van datamodellering is onderbouwd aan de hand van de verzamelingenleer in de wiskunde & relationele algebra. Als we een rechthoek schrijven met daarin employee: verzamelingen van alle x waarbij x een werknemer is. Je kan die populeren met concrete werknemers. Departement: verzameling van alle elementen die een departement zijn. Groepeer in verzamelingen die je een naam gaat geven en die zet je dan in het schema. Zo kan je gaan nadenken over alle mensen die rondlopen in de universiteit, groeperen in ’persoon’ of ’studenten’ en ’administratief personeel’,... Kan er overlap zijn? Je moet echt gaan redeneren in termen van verzamelingen. Level 0 en level 1: level 0 == objecten. Abstraheren naar verzamelingen: niveau 1. Level 2: denken in termen van welke concepten nodig zijn om een beschrijving te maken: wat is een entiteittype, wat is een relatietype,...

**Slide 15:** In de praktijk: je gaat je elementen oplsagen in een tabel. Uw employee, in plaats van te zeggen dat het een verzameling van elementen is, is het een tabel met daarin alle gegevens en per entiteit 1 rij. Elke rij is 1 element uit de verzameling. Namen van kolom zijn de attributen.

**Slide 16:** De eigenschappen, de kolommen.

**Slide 17:** Wat kan daarmee aan de hand zijn?

* Simpel vs complex (samengesteld: een adres heeft een straat, huisnummer, busnummer, postcode, stad en land).
* Single-valued vs multivalued: Excel met daarin de namen van een aantal mensen en kolom emailadres: meerdere emailadressen. Je hebt meerdere mogelijke waarden.
* Key attribute: attributen of combinaties van attributen die toelaten om entiteiten uniek te identificieren. Voornaam-achternaam is bv geen sleutel. Daarom heeft iedereen een studentennummer waarmee je dan uniek en zonder fout een student kan terugvinden. Die sleutelattributen zijn *de* manier om het juiste element eruit te halen.

**Slide 18:** Relatietypes: verbinden entiteiten. Je hebt relaties op niveau 0, de feitelijke relaties (relatie tussen ieder van ons en het vak) en je kan dat abstraheren naar relatietypes: is-docent-van, is-ingeschreven-voor → het abstracte begrip. Je moet de relatietypes een naam geven. Je kan die 1 naam geven, ook twee als je die in twee richtingen wilt lezen (doceert-vak en wordt-gedoceerddoor).

**Slide 19:** Graden van relaties:

* De meeste relaties zijn binair: tussen 2 verschillende entiteiten.
* Unaire relaties: van een entiteittype naar zichzelf. Bv: een persoon is baas van een persoon. Op toegepast level gaat dat niet naar zichzelf.
* Ternair: 3 entiteittypen: 3 ineens met elkaar verbinden. Beer - bought in - caf´e - by - customer. Je moet weten welk bier waar gekocht is door welke customer.

Associatietypes: in 2 richtingen: helpt om te verstaan wat de relatie feitelijk betekent → naam aan geven.

**Slide 20:** Cardinaliteit: het aantal objecten dat verbonden kan zijn met een ander object. Die getallen die bij de pijltjes staan. Als je bv van prof naar vak gaat, kan je zeggen wat het minimum en maximum aantal vakken is dat die prof kan doceren. Als niet moet: 0, anders 1, dan moet die dus minstens 1 vak doceren.

Het ER-model is een snapshot van 1 moment in de tijd en je probeert te beschrijven wat er altijd geldig is. Het kan zijn dat je in het verleden een relatie had met een persoon, maar dat er later geen lopende relatie meer is (bv vroeger leverde leverancier aan mij, maar nu niet meer). → Zet dan het minimum op 0. Je moet altijd redeneren van op elk ogenblik in de tijd. Elk aankooporder moet altijd een leverancier hebben, dus minimum 1. Een leverancier moet niet op elk ogenblik een order hebben lopen, dus minimum 0. Maximum: zelden een concreet getal: meestal n. op elk ogenblik in de tijd kan een docent meerdere vakken doceren.

**Slide 21:** Vb van aankooporder en leverancier. Elke leverancier heeft op elk ogenblik 0 tot meer aankooporders openstaan. Elk order heeft op elk moment in de tijd minimum en maximum 1 leverancier. → gebonden aan hoe de organisatie werkt, wat de regels zijn.

**Slide 22:** Elke werknemer moet altijd in een departement zitten (exact 1). Vanaf het moment van aanwerving *moet* je iemand in een departement zetten. Als het wel kan in je organisatie, moet je die 1..1 vervangen door 0..1. In Access: in de databanksoftware ga je zien dat de tabellen met elkaar verbonden worden met een lijntje → foreign keys. Je gaat tabellen met elkaar verbinden. Bij employee ga je het nummer opnemen van het departement waar de persoon zit. Doordat het 1..1 is, moet dat nummer altijd ingevuld worden.

**Slide 23:** Een student is ingeschreven in een opleidingsprogramma: minimum 1 of minimum 0 en voor elke student kan je zeggen in welke fase die zit. Thuisloze student: kan dat aan de KULeuven? Ben je dan nog student? Als je je niet meer inschrijft, ben je alle kenmerken van het student-zijn kwijt.

**Slide 24:** Komt ook op het examen!

* ER model geeft een snapshot weer. Je moet het lezen als ’is het op elk moment in de tijd waar dat...?’ Wat je niet kan zeggen, is uitspraken zoals ’een leverancier moet ooit een aankooporder hebben gehad’ → uitdrukkingskracht van het model is beperkt. Je kan niet gelijk welke constraint of business rule hiermee vatten. Bepaalde regels zijn niet heel duidelijk te tonen: bv werknemer heeft 0 departementen (zoals net aangenomen), maar als die wel een departement moet krijgen, moet je daar een proces rond maken zodat een business rule van bv binnen de 6 maand moet een departement toegewezen worden gemaakt worden, anders kan het volgens ER permanent 0 blijven. Je kan met bepaalde schema’s maar een deel vatten van wat gevat moet worden. Altijd meerdere schetsen nodig om iets volledig te documenteren.
* Omdat je regels kent, ben je geneigd om er dingen in te zien staan die er niet zijn. Als je 2 verschillende paden hebt, kan je denken dat ze bij hetzelfde uitkomen, maar dat is niet altijd zo. Bv je hebt een werknemer werkt in een departement en je hebt werknemer werkt aan een project en dat project is toegewezen aan het departement. Niets in het schema zegt dat het departement van de werknemer waar die werkt hetzelfde is als het departement van het project waar de werknemer aan werkt. Grondplan huis beschrijft ook niet het volledige huis! Als je verschillende paden hebt, kan je niet zorgen dat die leiden tot hetzelfde object, kan je niet afdwingen. Zie ook Figuur 12.1.

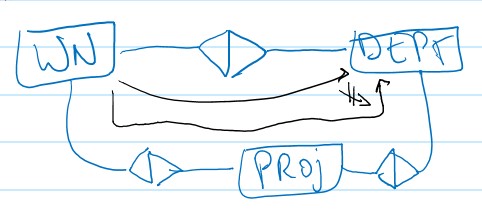


Figure 12.1: Duidende tekening bij Slide 24

**Slide 25:** Er zijn heel wat talen in omloop. Je hebt ook UML en Crow’s foot notation. Belangrijk is dat je je realiseert dat het onderliggend om dezelfde concepten gaat. UML is niet verschillend van ER. Je hebt de concepten van relaties en attributen geleerd. UML: n geschreven als \*. Crow: maximum meerdere: vogelpootje. Belangrijk voor ER: kunnen lezen. Niet kunnen maken. Beperkingen kennen (bv feit dat integriteit overheen verbanden niet kan worden afgedwongen. Uitspraken die verwijzen naar het verleden of de toekomst kunnen niet worden uitgedrukt. Examenvraag: ER-model met bepaalde uitspraken).

**Slide 26:** Normalisatie: van datamodel naar databank.

**Slide 27:** Je begint met een ER-model en je moet overgaan naar IT-ondersteuning en een databank ontwikkelen. *Niet* met requirements starten en onmiddelijk een databank ontwikkelen. Ook het ER-model niet direct omzetten naar een database. Je hebt ER-model, zet om naar relationeel model en dan naar databank (**Slide 28**).

**Slide 29:** Codd: data zo proper mogelijk hebben, met zo weinig mogelijk problemen: we moeten de hele verzameling aan attributen niet in 1 grote tabel steken, maar onderverdelen in deeltabellen. Heeft regels bepaald wat je moet samenzetten in 1 tabel en wat apart moet. Hoe beslis je wanneer iets in 1 tabel moet en wanneer niet (basis van relationeel model)? Omdat het relationeel model zo goed in elkaar zit, ge¨ımplementeerd is in veel databanksystemen. Dominant model voor databases van vandaag. Wat we moeten kunnen is SQL gebruiken om een relationele databank te ondervragen.

**Slide 30:** Tabel 6= tabel, maar relation wiskundige term.

**Slide 31:** Relationeel model gaat over het opsplitsen van tabellen in deeltabellen en wanneer is het goed? → Databank normaliseren! Wat is ”goed”/genormaliseerd? Je wilt redundancy vermijden, je wil proberen ten allen tijde vermijden dat je dezelfde feiten 2x opslaat. Je gaat ervoor zorgen dat je alle info maar 1x moet opslaan. Door het feit dat je alles maar 1x opslaat, ga je ook inconsistenties naar beneden halen. Als je bv zegt dat een bepaald studentennummer een bepaald persoon is en verderop staat dat hetzelfde nummer verwijst naar een ander persoon → inconsistentie. Regel: sla de link tussen studentennummer en naam maar 1 keer op.

Opbouw van zijn logica: kijken naar functionele afhankelijkheden tussen attribuuttypen en dan een aantal stappen doorvoeren met normaalvormen.

**Slide 32:** Waarom normaliseren? Veronderstel dat je zoals vanboven alles in 1 tabel zet, je ziet daar dan dat department number en department name 2 keer voorkomt (23 == finance). Je zou het maar 1 keer moeten opschrijven en niet herhalen, want stel dat finance verandert naar HRM, dan moet je dat op 2 plaatsen veranderen en als je dat niet doet, heb je inconsistente gegevens. Stel dat je een nieuw departement maakt, dan zou je een rij moeten toeveogen met 46 == ICT-department, maar je hebt daar nog geen werknemer voor → halve rij → ineffici¨ent. Wat als Lee Wong het bedrijf verlaat? → rij schrappen, maar ook kwijt dat 38 R&D is. Je gaat dus alles wat te maken heeft met werknemers in 1 tabel zetten en alles wat te maken heeft met departementen in een andere tabel. Maak gebruik van kruisverwijzingen (cross references). Op die manier zijn links maar 1 keer opgeslagen. Je gaat ook niet zitten met halfvolle rijen.

**Slide 33:** Kruisverwijzingen: foreign key. Dit laat je toe om te navigeren doorheen de tabellen. Het basisprincipe dat Codd wiskundig heeft aangetoond is dat als je van een grote tabel vertrekt en je die normaliseert, er geen informatie verloren gaat. Als je door middel van kruisverwijzingen de eerste tabel terug zou afleiden, ben je geen informatie kwijt.

**Slide 34:** Verloop normalisatie: vertrek van alle attributen samen. Naar 1NF (normaalvorm), 2NF, 3NF. Die buiten de rechthoek moeten we niet meer kennen.

**Slide 35:** 1NF: alle attributen moeten een enkelvoudige waarde zijn. Composite en multivalued attributen: daarvanaf geraken.

2e en 3e: duplicatie en onderhoud en je gaat zo dan uw tabellen splitsen om dat te optimaliseren.

**Slide 36:** Basis is functionele afhankelijkheid: waarde van 1 attribuut wordt bepaald door de naam van een ander attribuut. Als je het studentennummer kent, weet je de naam, adres, geboortedatum,.. → Relatie tussen sleutel- en andere attributen. Basisprincipe: een functionele afhankelijkheid maar 1 keer opslagen → die dingen samenzetten in 1 tabel.

**Slide 37:** Stel dat je punten gaat opschrijven. Je schrijft op voor welk vak, welk studentennummer en dan geef je de naam van de student en de bijhorende grade. Dan zondig je tegen het principe van 2NF: die zegt dat alles afhankelijk moet zijn van de sleutel. Student name is afhankelijk van student ID, dus apart zetten. En dan course ID en student ID, die geven de punten. Student name is slechts afhankelijk van een deel van de sleutel (denk ik).

3NF: geen transitieve afhankelijkheden. Website is gebonden aan de faculteit en niet rechtstreeks aan de prof. Bij de prof enkel faculteit overhouden als foreign key naar faculty en facultywebsite. Wat moet je kunnen doen? Op het examen: lijst van functionele afhankelijkheden en dan wordt zo een relatie gegeven zoals op Slide 37 en dan moet je zeggen in welke NF de relatie staat: kunnen beoordelen of het waar/fout is dat het in zoveelste NF staat. 1NF:

2NF: sleutel met meerdere attributen: al wat volgt moet afhankelijk zijn van de volledige sleutel en niet van een deel ervan.

3NF: geen ketting van functionele afhankelijkheden. Enkel afhankelijk van sleutel en niet van iets anders. Anders in aparte tabel zetten met behulp van foreign key.

**Slide 38:** Belang van NF: heel veel bedrijven gebruiken een of ander ERPsysteem en geven gebruikers mogelijkheid om een Excel-sheet te downloaden die niet genormaliseerd gaat zijn: tabel met vanalles samengegooid. Als je die moet gebruiken voor verdere verwerking, heb je een probleem. Vandaar is het goed dat je zelf beslist of het misschien nodig is dat je die opsplitst, duplicaten weghaalt,... → praktisch gebruik.

SQL video niet bekijken. Enkel over ER en normalisatie. Er staan examenvragen online die we vrijdag samen gaan oplossen.

**Chapter 13**

# ER Model vLes 12

## 13.1 Slides: 4A Information Modeling ER Modelling

**13.1.1 Video: Business Information Systems 31: Informa-**

### tion Management: ER Model [Dutch]

**Slide 2:** We gaan kijken naar de SCOPE layer binnen Zachman. We gaan kijken wat de informatiebehoeften zijn van een bepaalde organisatie. De bedoeling is om die informatiebehoefte zo compleet mogelijk in kaart te brengen. We gaan proberen een zo exhaustieve opsomming van de gegevens te geven en de verbanden daartussen binnen een bepaalde context te duiden; zeg maar binnen een bepaald bedrijfsproces. Ook proberen we te verhinderen om overtollige informatie te vermijden. Bedoeling is om de scope so nauwkeurig mogelijk af te bakenen. We wensen deze compleet af te bakenen zonder redundante gegevens mee op te nemen.

**Slide 3:** We gaan dat doen door verschillende bedrijfsconcepten te defini¨eren. Hangt af van de context! Bij administratiesysteem: klant: wat is dat? Wat zijn de karakteristieken van een klant? → attribuuttypen: klantnummer, -naam en -adres. Ook ander concept (entiteit) zoals bv product. We gaan alle concepten oplijsten, modelleren en zoeken naar de verbanden. De bedoeling is om een conceptueel gegevensmodel te cre¨eren dat gaat focussen op de concepten. Kan een ER-model zijn, ook EER en UML.

**Slide 4:** Vb eenvoudig ER-model. Student = entiteittype: concept waarrond wij gegevens wensen te verzamelen (attribuuttypen). Studieprogramma is ook een entiteittype waarrond we gegevens wensen te verzamelen. Er bestaat een verband tussen deze 2 entiteittypen: student kan minstens 0, maximum 1 studieprogramma’s volgen. Je ziet dat boven het verband getallen staan aangegeven: minimum- & maximumcardinaliteit. Minimumcardinaliteit kan 0 of 1 zijn, die is opgelegd door het bedrijf: het is misschien niet nodig dat studenten ingeschreven zijn voor een studieprogramma. Wordt u opgelegd door de bedrijfsomgeving, door de karakteristieken van het proces. Je kan ook gaan kijken op niveau 0 naar een aantal entiteiten: concrete voorkomens van een entiteittype: je gaat kijken naar specifieke studenten en study programs. Op niveau 0 ga je kijken naar concrete voorkomens van de entiteittypen.

**Slide 5:** Dit moet allemaal ondersteund worden door uw informatiesysteem en die moet allerlei types gaan aanbieden:

* Storage, opslag: uw IS zal moeten toelaten om uw ER-model op te slaan onder de vorm van een database zodat je gegevens kan verzamelen omtrent studenten en studieprogramma’s.
* Je moet het niet alleen kunnen opslaan, maar ook verwerken. Je kan het gaan bevragen door middel van SQL → gegevensmodel bevragen.
* IS zal ook de kwaliteit van de informatiediensten moeten verzorgen: inputkwaliteit: moet gemakkelijk gebeuren, compleet zijn. Alle gegevens van een student moeten ingegeven worden (compleet zijn) en die input moet accuraat zijn. Ook zorgen dat het correct verwerkt wordt, dat het betrouwbaar is: als je de punten van een student wil opvragen, moet je zorgen dat die punten correct verwerkt worden. Verwerking en analyse moet volledig correct kunnen gebeuren. Output moet tijdig kunnen gebeuren: tijdig genereren en zorgen dat die de juiste selectiviteit heeft: het juiste type informatie voor de juiste doelgroep en dat die op de juiste manier afgeleverd wordt.

**Slide 6:** We gaan kijken naar de requirements analysis. Kijken wat de gegevensbehoeften zijn en hoe we die zo accuraat mogelijk in kaart kunnen brengen.

**Slide 9:** Neem het voorbeeld van een bedrijf zoals Callebaut. Heeft verschillende processen om de chocolade te maken: bonen aankopen, suiker,... We gaan kijken naar zo’n aankoop- & ontvangstsysteem en wat de processen zijn die moeten gebeuren. Allereerst moet de beslissing genomen worden om de bonen te kopen. Gaan we aankopen bij een leverancier (gaan we moeten selecteren: ook al een belangrijk proces). Cacaobonen kunnen geleverd worden en de kwaliteit moet onderzocht worden: van juiste kwaliteit? Geleverd in de juiste hoeveelheid & op tijd? → Allemaal processen bij de levering. Daar moeten verschillende types van gegevens verzameld worden (productnummer & -naam). Ook verschillende suppliers mogelijk die elk hun eigen nummer en naam hebben. Purchase price: zal wellicht afhangen van het type cacaobonen en het type supplier. De leveringstermijn is een ander belangrijk informatie-element dat ook zal afhangen van type bonen & leverancier. In dat proces zijn er verschillende stappen en gegevensbehoeften. Bedoeling is om dat ganse proces netjes in kaart te brengen.

**Slide 10:** Nog steeds voor Callebaut: het nauwkeurig in kaart brengen van informatiebehoeften is niet makkelijk want je moet met heel wat mensen praten, heel wat processen inspecteren en vaak word je overspoeld door irrelevante, oninteressante informatie. Veel mensen zijn slechts een schakel in heel het proces en hebben dus slechts een beperkt inzicht in het proces. Vaak zie je ook dat indien je met bepaalde mensen spreekt, sommigen te gedetailleerde informatie geven die niet zinvol is bij het opstarten van uw proces, anderen geven te geaggregeerde, te algemene info. De bedoeling van een informatie-analyst of een informatiearchitect is om een brug te vormen tussen de business en de IT. IT: mensen die de database bouwen. Business: mensen die zich bezighouden met het controleren van de aankoop, kwaliteit,... Als informatieanalyst/-architect ben je een perfecte brug tussen deze 2 partijen: in staat zijn om de business te begrijpen, om je onder te dompelen in het bedrijfsproces, wat er allemaal gebeurt, maar ook in staat zijn om al die zaken die er gebeuren in kaart te brengen in de vorm van een informatiemodel dat kan doorgespeeld worden aan de databaseontwikkelaar die dat kan implementeren. Je bent informatiearchitect. De architect van een huis vervult dezelfde functie: gaat kijken naar de bouwheer (welk huis moet gezet worden?), plan maken en dat geven aan onderaannemers. Infomartieanalyst/-architect doet exact hetzelfde maar bouwt een plan van een IS. Je kunt dat plan van verschillende kanten gaan bekijken (data- of procesperspectief). In dit hoofdstuk: dataperspectief.

(Tip voor examen: info-analyst/-architect is een programmeur: waar/vals → Vals! Gaat info analyseren/architectureel plan van opstellen zodanig dat dat kan ge¨ımplementeerd worden door een programmeur.)

**Slide 11:** Informatieanalyst gaat heel nauw moeten samenwerken met de eindgebruiker (aankoopmanager, inspectiemanager, mensen van de dienst facturatie,...). Je moet die mensen gaan bevragen (niet simpel want mensen gaan altijd naar het bestaande systeem verwijzen. Mensen zijn slecht in het out-of-the-box thinking). Een andere belangrijke uitdaging is dat men vaak gaat refereren naar recente informatiebehoeften (men had iets nodig, ontbrak → gefrustreerd), gaat men veel belang aan hechten; hoewel het bij de initi¨ele uitbouw van het IS misschien helemaal niet zo belangrijk was. Heel wat van die business/end users zijn slechts een schakel in een gans systeem. Daardoor hebben ze een beperkt zicht op de totale informatiebehoefte en kunnen zich heel moeilijk loskoppelen van het bestaande systeem.

**Slide 12:** Technieken voor requirementsanalyse:

* Observatie: onderdompelen in bedrijfsproces en kijken wat er allemaal gebeurt. Wordt daadwerkelijk gebruikt: men gaat een aantal informatieanalysten op de werkvloer laten rondlopen en de bedrijfsprocessen laten observeren.
* Interviewen: bepaalde mensen interviewen en vragen wat er precies gebeurt en nodig is. Met welke gegevens wenst u te werken?
* Surveys: al dan niet anoniem (voordeel: eens goed mening spuien en wat er echt zou moeten verbeteren aan het huidige bedrijfsproces).
* Documentanalyse: wat zijn de documenten (bv aankoopbon, inspectiebon [als cacaobonen binnenkomen]) en je gaat ze inspecteren, wat er allemaal opstaat en hoe die worden ingevuld. Zijn er velden die vaak opengelaten worden/manueel toegevoegd worden?
* Brainstorming: out-of-the-box thinking: hoe zouden de zaken eruit zien indien u het voor het zeggen had? Een manier is de delphimethode: gestructureerde manier van vragen stellen om een consencus te bereiken rond een bepaald probleem. Ook decision analysis techniques.
* Prototype: interface bouwen. Systeem zonder functionaliteit. Geven aan de klant, zodat die kan beoordelen of het systeem al dan niet ideaal is voor hem/haar.

→ Combineren!

**Slide 13:** Data modeling (definitie niet vanbuiten kennen). Laat u toe om de semantiek, om de universe of discourse (bedrijfscontext) vast te leggen, zo accuraat mogelijk, zonder u zorgen te maken omtrent een bepaalde implementatie. Deze vraag is hier niet aan de orde. Wel het conceptueel gegevensmodel zo goed mogelijk opstellen. Zo nauwkeurig mogelijk in kaart brengen.

Belangrijk model: ER-model: kijken naar entiteittypen en relationshiptypen. Later zal dit moeten ge¨ımplementeerd worden, maar dat gebeurt in een later stadium. In dit stadium is het enige dat belangrijk is, het opstellen van het ER-model.

(Examenvraag: ER model is implementatie-specifiek model. → FOUT!)

**Slide 14:** ER-model: formalisme om een conceptueel gegevensmodel te bouwen. Zal een realistisch beeld van het bedrijfsproces bieden op een formele manier. Zal een bedrijfsproces heel nauwkeurig in kaart brengen. Bedoeling: compromis vinden tussen expressiviteit en begrijpbaarheid: ER-model is brug tussen eindgebruiker en IT’ers (formeel genoeg). Grafisch voorgesteld door middel van een ER-diagram.

**Slide 15:** Vb ER-model: je hebt 3 entiteittypen: supplier, product en purchase order. Zijn concepten rond dewelke wij gegevens wensen te verzamelen. Supplier = entiteittype, Peter Jansen is een entiteit. → Onderscheid tussen het type en het individueel voorkomen. De elementen in ellipsen zijn attribuuttypen. Individueel voorkomen van een attribuuttype is een attribuut of een attribuutwaarde. Bepaalde attribuuttypen zijn onderlijnd: sleutelattribuuttypen: attribuuttype waarmee je uniek entiteiten gaat kunnen identificeren (supplier heeft bv een uniek suppliernummer). Naast de entiteittypen & attribuuttypen: relationshiptypen: laten ons toe om verbanden te leggen.

Tussen supplier & product: verband = bv levert: supplier levert producten → kan in 2 richtingen bekeken worden. Ruit die eigenlijk uit 2 pijlen bestaat en de rollen aanduidt. Dat verband wordt ook gekenmerkt door de getallen die in beide gevallen 0 tot n zijn: minimum- & maximumcardinaliteit. Supplier kan bv minstens 0, maximum n producten supplyen. Je ziet dat dit relationshiptype bepaalde attribuuttypen heeft. Je hebt attribuuttypen bij deze relatie (nl deliv period en purchase price). Je kan deze niet apart laten voorkomen, om die attribuuttypen te kennen moet je weten over welke leverancier het gaat en welk product het is. Om de levertermijn te kennen moet je zowel supplier als product kennen.

Minstens 1 & maximum 1 supplier: exact 1 supplier: een aankooporder kan minstens 1, maximum n producten hebben. Product kan in bestelling zijn in minstens 0, en maximaal n orders. Quantity: hoeveelheid van een bepaald product in een bepaald model. Je kan je zo verschillende vragen stellen: waaromtrent wenst u gegevens te verzamelen? Verbanden tekenen,...

**Slide 16:** Entiteit is een object in de re¨ele wereld, dat onafhankelijk kan bestaan. Kan fysisch zijn (klant, leverancier,...), maar ook conceptueel (job,...). Elke entiteit heeft bepaalde karakteristieken die we attributen noemen. Een attribuut is ook in 2 delen opgesplitst: attribuuttype is bv naam en een specifieke instantie ervan (bv Jan Jansen) is dan een attribuut.

**Slide 17:** Sommige waarden kunnen null zijn: onbekend/niet van toepassing. Kan ook zijn dat sommige studenten nog geen (email)adres hebben: zal null zijn.

* Type
* Instantie: individueel voorkomen van type.

Classificatie vs instantiatie: LU en Callebaut zijn chocolademakers (kan je zo classificeren), instantiatie: starten van chocolademaker en dat instanti¨eren (instanties van maken) zoals LU en Callebaut. Entiteittype defini¨eert een verzameling die dezelfde karakteristieken hanteren.

**Slide 18:** Modelleren: altijd op niveau 1: kijken naar entiteittypen.

**Slide 19:** Access: relationele databasetoepassing die u toelaat om ER-modellen te modelleren.

**Slide 20:** Composite vs simple (atomic):

* Composite: samengesteld. Naam is bv samengesteld: voornaam & familienaam.
* Simple atomic: niet samengesteld. Bv salaris.

Wat maakt of iets samengesteld is of niet? Het is terug de business die dat gaat bepalen: willen ze voor- & achternaam apart of samen?

* Enkelwaardig: je hebt er maar 1 waarde voor (bv leeftijd).
* Meerwaardig: meerdere waarden mogelijk (bv diploma’s, emailadressen).

Key attribute type: laten toe om entiteiten op een unieke manier te identificeren. Bv productnummer, employee: social security number → voor 1 waarde van het attribuuttype heb ik precies 1 waarde. Het kan ook zijn dat voor een bepaald attribuuttype je meerdere attribuuttypen moet combineren om tot een sleutel te komen. Bv vlucht: vluchtnummer & datum vormen samen sleuteltype → hangt ook weer af van het bedrijf. Sleutelattribuuttype heeft uniqueness constraint: moet een unieke waarde hebben.

**Slide 21:** Sleutelattributen in access.

**Slide 22:** Relationshiptype: gaat homogene verbanden, associaties cre¨eren tussen entiteiten.

* Relationship: individueel voorkomen van associatie, verband.
* Relationship type: algemene heeft een naam.

Attribuuttypen bij relationshiptypes: attribuuttypen die enkel gekend zijn indien men beide entiteiten van het relationship type kent.

**Slide 23:** Degree: aantal entiteittypen dat deelneemt in het relationship type: • Unair: relationship type met 1 entiteittype.

* Binair: 2 entiteittypen participeren, bv supplier & product → altijd 2 richtingen volgens dewelke je kan gaan kijken.
* Ternair: 3 participerende entiteittypen, iets minder frequent.

**Slide 24:** Cardinaliteitsratio: gaat het minimum & maximum aantal keer weergeven dat een bepaalde enteit kan gekoppeld zijn aan een andere binnen het relationshiptype. Minimum:

* Indien 0: kan voorkomen zonder gekoppeld te zijn aan andere een entiteit → parti¨ele participatie.
* Indien 1: entiteit moet altijd geconnecteerd worden met minstens 1 andere entiteit (purchase order moet altijd geconnecteerd zijn met een supplier) → bestaansafhankelijkheid (existence dependency).

Maximum:

* 1: entiteit is geconnecteerd met ten hoogste 1 andere entiteit (purchase order kan niet met meer dan 1 supplier geconnecteerd zijn).
* n: supplier kan bv wel meerdere purchase orders hebben staan.

**Slide 25:** Relationship types: on order: supplier en purchase order: relationship type van graad 2. supplier heeft minstens 0 en maximaal n purchase orders. Purchase order heeft mininmaal & maximaal 1 supplier.

**Slide 26:** Vb ander relationship type: werkt in: employee werkt in minstens 1 en maximaal 1 departement (dus exact 1). In een departement werken minstens 0, maximaal n employees. Merk op: minimaal 1 wijst altijd op bestaansafhankelijkheid: employee is bestaansafhankelijk van een departement.

**Slide 27:** Relationship type van graad 1: met maar 1 entiteit, bv: hi¨erarchische verantwoordelijkheden employees: employee gaat andere werknemers superviseren, maar een werknemer wordt ook gesuperviseerd door een andere werknemer. De hi¨erarachische verantwoordelijkheden geven weer welke werknemer welke andere gaat superviseren. Kan je ook gaan bekijken in termen van cardinaliteit: werknemer superviseert minimum 0 en maximum n employees en een employee wordt door minstens 0, maximaal 1 andere employee gesuperviseerd. Je kunt het recursief relationship type van graad 1 dus in 2 richtingen gaan lezen. De min- & maxcardinaliteit haal je weer uit de business. Vb: werknemers die meerdere andere werknemers superviseren, maar ook mensen die niemand superviseren. Je ziet ook dat iemand gesuperviseerd wordt door soms 0, max n andere werknemers.

**Slide 28:** Binnen entiteittypes moeten we een verder onderscheid maken tussen sterke en zwakke entiteittypen:

• Zwak: entiteittype dat geen eigen sleutel kan voortbrengen.

**Slide 29:** Bv: hotel met naam en room heeft room nr. Hotel kan 0 kamers hebben (hotel in opbouw dat nog geen kamers heeft, maar je wil wel al gegevens opslagen over het hotel). Hotel kan ook meerdere kamers hebben → n als maximum. Een kamer behoort aan minstens 1 en maximaal 1 hotel toe (→ bestaansafhankelijkheid). Sleutelattribuuttypen: voor hotel: hotel naam. Voor kamer: kamer nr: laat dat je toe om een kamer uniek te identificeren? Nee, want je hebt kamernummer 100 in Brussel EN in Leuven. → Kamer is een zwak entiteittype dus je moet de hotel naam erbij gaan zetten. Room nr is een parti¨ele sleutel. Indien je de hotel naam erbij aan toevoegt + kamer nr, kan je wel uniek een kamer gaan identificeren.

Zwak entiteittype is altijd bestaansafhankelijk! Is een bestaansafhankelijk entiteittype altijd zwak? (links): supplier hefet supplier nr en purchase order heeft ponr. Purchase order is bestaansafhankelijk van supplier, maar is niet zwak want purchase order heeft zijn eigen ponr! → Bestaansafhankelijk entiteittype dat niet zwak is. ⇒ Zwak entiteittype is altijd bestaansafhankelijk, bestaansafhankelijk entiteittype is niet altijd zwak!

Wat maakt nu dat een entiteittype zwak is? De business beslist daarover! Stel dat de kamernummers uniek (over alle ketens) ge¨ıdentificeerd werden, was kamer niet zwak.

**Slide 30:** Vb ER-model uitgewerkt voor werknemersadministratie: we hebben employee en recursief relationship type supervises. Employee heeft ename, ssn (sleutelattribuuttype) en eaddress. Zo ook voor anderen. Verschillende relationship types: ER geeft heel duidelijk wat de verschillende verbanden zijn en wat informatie is.

**Slide 31:** Tekortkomingen:

* Laat niet toe om temporele verbanden weer te geven: bv employee, als die aangeworven wordt, laten we die in verschillende departementen een stage lopen om dan na 6 maand een ”permanent” departement te geven

→ kan je niet modelleren. Ander vb: opleggen dat een employee niet mag terugkeren naar een departement waar hij voorheen manager van was.

* Laat niet toe om business rules/integriteitsregels op te leggen die meerdere relationshiptypes omspannen. Bv: tussen employee en departement heb je het relationshiptype works in en manages. Uiteraard bestaat daar een logisch verband tussen want als een employee in het departement marketing manager is, moet hij daar ook werken. Het kan niet zijn dat employee x manager is van marketing, maar werkt in finance. Volgens het getoonde ER model is dat perfect mogelijk. Ander vb: 3 relationshiptypes vanuit employee: naar project, naar departement en van employee naar zichzelf. Daar heb je ook terug een regel die al die relationshiptypes gaat connecteren: het moet zo zijn dat een employee enkel kan werken aan departementen waar die employee werkt. Kunnen we binnen ER weer niet afdwingen. Je kan niet gaan afdwingen, modelmatig, dat een werknemer enkel mag werken aan projecten uitgevoerd door het departement waar hij tewerkgesteld is.

**Slide 32:** Samenvatting van notatie. Zwak entiteittype: dubbele lijn.

**Slide 33:** Ontwikkelen: entiteittypen zoeken. Makkelijke vuistregel: kijk naar zelfstandige naamwoorden, relationship types vastleggen, rollen, cardinaliteiten (altijd in samenspraak met de business!), attribuuttypen bepalen en verbinden, sleutelattribuut typen bepalen.

## 13.2 Slides: 4A Information Modeling ER Modelling

**13.2.1 Video: Business Information Systems 32: Informa-**

### tion Management: EER Model [Dutch]

**Slide 39:** ER uitbreiden om nog meer van de semantiek van de universe of discourse in kaart te brengen. EER-model gaat bijkomende modelleringsconcepten aanrijken: generalisatie/specialisatie, categorisatie en aggregatie.

**Slide 40:** Generalisatie/specialisatie: vb van studentenadministratie: welke entiteittypen gaan we modelleren? Prof, student en cursus. Kijk naar prof & student: welke attribuuttypen horen daarbij? Bij prof: profnummer, naam, -geslacht,... analoog bij student: attribuuttypen zoals studentnummer, -naam,... Als je kijkt naar de attribuuttypen daartussen, worden er veel gedeeld. Wat we kunnen gaan doen is het entiteittype prof & student generaliseren in een entiteittype ”persoon”, die dan attribuuttypen zoals geslacht, naam,... gaat hebben. Prof en student zijn dan subklassen van persoon. Van prof & student naar persoon is een generalisatie, omgekeerd is een specialisatie. In heel wat real-life voorbeelden kan je een klasse specialiseren in verschillende subklassen: subgroep van entiteittypen die we afzonderlijk gaan beschouwen omdat ze speciale karakteristieken hebben. Inheritance/overerving: een subklasse erft alle karakteristieken over van de superklassen (attribuuttypen, maar dus ook relationshiptypen).

**Slide 41:** Ander vb: werknemersadministratie: superklasse werknemer, subklasse salaried employee, hourly employee en contracted employee. Voordelen: bij specialisatie kun je voor bepaalde subgroepen specifieke karakteristieken toevoegen (specifieke attribuuttypen en relationshiptypen). Zeker wanneer bepaalde relationshiptypes enkel bestaan voor bepaalde subgroepen is dit een zeer zinvolle optie.

**Slide 42:** Specialisatie: top-down: van persoon naar student en prof: specialiseren. Generalisatie is het omgekeerde: bottom-up synthese. Overerving speelt hierbij een belangrijke rol: subklassen erven zowel attribuuttypen als relationshiptypen.

**Slide 43:** Van specialisatie/generalisatie kun je bijkomende karakteristieken gaan vastleggen:

* Disjointness: gaat specificeren of dat een entiteit tot beide subklassen tegelijk kan behoren. Als je een superklasse persoon hebt met 2 subklassen, student en prof, kan je met disjointness vastleggen of die tot beide subklassen mogen behoren. In dit vb waarschijnlijk niet, dus disjoint: entiteit kan ten hoogste tot 1 subklasse behoren, niet tot beiden tegelijk. Overlap: je kan specialisaties hebben waarbij een entiteit tot beide subklassen kan behoren.
* Completeness constraint: onderscheid maken tussen totale en parti¨ele specialisatie: totaal: entiteit behoort altijd tot 1 van de subklassen. Bij persoon: alle personen zijn ofwel prof, ofwel student. Parti¨eel: entiteit kan een persoon zijn, maar geen student of prof. Je hebt entiteiten die tot de superklasse behoren, maar niet tot een subklasse.

Business bepaalt weer wat de karakteristieken zijn van uw generalisatie/specialisatie.

**Slide 44** Voorbeeld: specialisatie van product in afgewerkt product en assembly product. Specialisatie is totaal: alle producten zijn ofwel afgewerkt, ofwel assemblageproducten. De specialisatie is disjoint: product is ofwel finished, ofwel assembly, niet beiden tegelijk. In deze specialisatie gaan de subklassen bepaalde karakteristieken/relatie toevoegen.

**Slide 45:** Parti¨ele specialisatie en overlap. Artiest is onderverdeeld in painter en sculptor, maar er zijn artiesten die niet tot een van de subklassen horen. Overlap: er kunnen artiesten zijn die zowel painter als sculptor zijn.

**Slide 46:** Bemerk dat eenzelfde entiteittype het voorwerp kan zijn van meerdere specialisaties, zoals in het gegeven vb: employee: wordt eerst gespecialiseerd in secretaris, ingenieur en technicus → totaal en disjunt: alle employees behoren tot exact 1 van deze subklassen. Daarnaast is employee ook het voorwerp van een parti¨ele specialisatie: manager: subset van employees is manager. Daarnaast ook totale disjuncte specialisatie in permanente en tijdelijke employee.

**Slide 47:**

* Specialisation hierarchy: elke subklasse heeft ten hoogste 1 superklasse. Elke werknemer heeft ten hoogste 1 baas bv.
* Specialisation lattice (specialisatieraster): daar kun je wel hebben dat een subklasse meer dan 1 superklasse heeft. Subklasse gaat dus ook van meerdere superklassen overerven → multiple inheritance.

**Slide 48:** Specialisation lattice: meervoudige overerving: persoon die totale overlapspecialisatie heeft tussen employee en student (kan dus beiden tegelijk zijn). Employee wordt verder gespecialiseerd in staff, faculty en student assistant. Student wordt verder gespecialiseerd in student assistant, graduate student en undergraduate student. → specialisatieraster want student assistant is subklasse van zowel employee als student, vandaar ook totale overlap bij person. Kan aanleiding geven tot ambigu¨ıteit: stel dat student en employee 2 attribuuttypen hebben met dezelfde naam, dan zullen die 2 attributtypen botsen in de student assistant subklasse: wordt daar belangrijk om een goed onderscheid te kunnen maken tussen die 2 attributtypen.

**Slide 49:** Categorisatie: superklasse/subklasse relatie, waarbij je meer dan 1 superklasse gaat hebben en waarbij de superklasse weer verschillende entiteittypen gaat voorstellen. De subklasse in het vb zal dan een verzameling (van entiteiten) zijn die een subset is van de unie van de entiteiten van de superklasse. We noemen die subklassen een union type, of categorie.

**Slide 50:** De genoemde zaken worden ge¨ıllustreerd met volgend vb.

**Slide 51:** 2 superklassen: natural person (bv Jan Peters en Peter Janssen) en legal person (IBM, Microsoft). Account holder is dan een subset van de unie van de superklassen: in account holder kun je Jan Peters hebben en IBM. Zijn een deelverzameling van de unie van de superklassen. → Vb van een parti¨ele specialisatie: niet alle entiteiten van de superklassen behoren tot de subklasse. De categorisatie kan ook totaal zijn: alle entiteiten van de superklassen behoren tot de subklassen (dus Jan Peter, Peter Janssen, IBM en MSFT zijn account holders). Wat is nu het onderscheid tussen dit en specialisatie/generalisatie? Bij specialisatie/generalisatie kun je altijd de ”is een”-test doen: ”prof *is een* persoon”. We gaan kijken naar de categorisatie. Indien die parti¨eel is, kunnen we natuurlijke persoon een subklasse maken van account holder. Indien de specialisatie parti¨eel is, kan dat niet want in natuurlijke persoon zaten Jan Peters en Peter Janssens, maar in account holder zat enkel Jan Peters. ”Is een” gaat hier dus niet (altijd) op. Kunnen we account holder een subklasse maken van natural person of legal person? Nee want in account holder zit bv IBM, is geen natural person. Indien de categorisatie parti¨eel is, gaat specialisatie/generalisatie dus niet. Indien de categorisatie totaal zou zijn (alle natuurlijke personen en alle legal personen zijn account holders), dan kunnen we wel natuurlijke persoon en legal persoon een subklasse maken van account holder. Samengevat: indien categorisatie parti¨eel is: geen generalisatie/specialisatie mogelijk. Indien totaal: wel.

Selectieve overerving: entiteit erft over afhankelijk van zijn afkomst. Bv: ofwel van natuurlijke persoon ofwel van legal persoon. Komt niet zo vaak voor in de realiteit, maar kan universe of discourse wel krachtig weergeven in het EERmodel.

**Slide 52:** Aggregatie: zaken samenvoegen: composieten bouwen: object dat samengesteld is uit andere objecten. Het composiet object kan op zijn beurt bepaalde attribuuttypen hebben, of bepaalde relationshiptypes hebben.

**Slide 53:** Job en applicants: we hebben die 2 gegroepeerd in een aggregaat interview. Op dat niveau kunnen we een attribuuttype toevoegen, bv duurtijd (van interview). Je kan ook relationshiptypes gaan toevoegen, is hier ook gebeurd: relationshiptype die relatie legt met job offer. Bij een aggregatie ga je altijd entiteittypen en relatietypen samenvoegen in een soort hoger-niveau constructies. Op dat niveau kan je dan attribuuttypen en relationshiptypen bijvoegen.

**Slide 54:** Eerder vb van EER-model met de concepten hierboven beschreven toegevoegd: employee: specialisatie toegevoegd (parti¨eel). Aggregatie is ook uitgevoerd: departement en project geaggregeerd in assignment: toewijzing van een project aan een bepaald departement. We hebben op het niveau van het aggregaat een bijkoment relationship type gemaakt: works on.

Aan de hand van modelleringsconcepten kun je meer semantiek aan het model toevoegen. Maar bemerk dat de 2 essenti¨ele tekortkomingen van bij ER ook hier nog gelden: geen temporele verbanden & als je businessconstraints hebt, integriteitsregels die meerdere relationship types gaat omspannen, dit hier ook niet weergegeven kan worden.

## 13.3 Slides:4B Information Modeling RelationalModel

**13.3.1 Video: Business Information Systems 33: Informa-**

### tion Management: Relational Model [Dutch]

Let op: de slidenummers uit de PPT komen niet overeen met die hier. Mijn slidenummers komen overeen met de gegeven PPT vanop Toledo.

**Slide 2:** Relationeel model == implementatiemodel: zal gebruikt worden om een database te implementeren. Je start van de gebruiker en van die gebruiker gaat de informatiearchitect de behoeften verzamelen aan de hand van een ER of EER model. Dat E(E)R model zal vertaald worden naar een implementatiemodel: het relationeel model. Dat zal dan ge¨ımplementeerd worden onder de vorm van een database.

Realtioneel model: zeer formeel en wiskundig onderbouwd: niet geschikt om de behoeften van de gebruiker vast te leggen (leunt dichter aan bij implementatie). ER-model kan ook niet rechtstreeks ge¨ımplementeerd worden. Omvormen naar relationeel model is niet wat een informatiearchitect doet, maar het is zeer belangrijk te weten wat de programmeur doet.

**Slide 4:** Relationeel model: oorspronkelijk ontwikkeld door Codd, is een formeel model met stevige wiskundige onderbouwing, gebaseerd op verzamelingenleer. Is gebaseerd op allerlei onderliggende theorie¨en, heeft geen grafische representatie (daarom ook geen ideaal communicatieinstrument naar de gebruiker toe). Niet voor requirements. Niet ge¨ımplementeerd op zich, maar wel in Access bv. Databasetoepassing gaat typisch ook een taal voorzien, SQL, die je toelaat om databases te bevragen op een gestructureerde manier.

**Slide 5:** Relationeel model heeft 2 belangrijke constructoren:

* Tuple-constructor (== entiteit) → tuples aanmaken via het aggregeren van attributen. Die tuple-constructor kan enkel toegepast worden op atomaire attribuuttypen. Composiete attribuuttypen worden hier dus niet ondersteund. Relatie (== entiteittype).
* Set-constructor: kan enkel toegepast worden op tuples, niet op attributen. Betekent dat er dus geen meerwaardige attribuuttypen ondersteund worden (bv student die meerdere emailadressen kan hebben).

→ Complexe objecten kunnen niet rechtstreeks gemodelleerd worden in het RM → trukjes voorzien om dit toch weer te geven.

**Slide 6:** Relatie == tabel van waarden, elke rij komt overeen met een tuple. De attributen worden weergegeven in de kolommen, die ook elk een naam hebben. De waarde van een attribuut komt overeen met specifieke cel.

**Slide 8:** Een relatie gaat overeenkomen met een entiteittype in het ER-model en specifi¨eert het object waarover we gegevens wensen te verzamelen. Een relatie is dus eigenlijk een verzameling van tuples. Er zitten dus geen duplicaten in en de tuples hebben ook geen volgorde (want willekeurig gesorteerd in tabel). Als je kijkt naar een bepaalde relatie, zijn er evenveel tuples aanwezig als dat er employees aanwezig zijn in de tijd. Een relatie kan je ook mathematisch karakteriseren: subset van het carthesisch product van verschillende domeinen (staat voor het waardebereik, alle mogelijke waarden voor een bepaald attribuut). Bv productnummer met waarden tussen 1 en 999.

Domein: range van attribuut in kaart brengen. Het domein kan atomair zijn (niet verder op te splitsen zonder zijn waarde te verliezen) of samengesteld: kan wel opgesplitst worden op een zinvolle manier.

**Slide 12:** Vb carthesisch product: geeft alle mogelijke combinaties weer. De relatie die je in de werkelijkheid ziet, zal een subset zijn van alle mogelijke combinaties.

**Slide 9:** Attributen: elk attribuut is gedefini¨eerd volgens een bepaald domein die het waardebereik ervan gaat weergeven. Bemerk dat eenzelfde domein meerdere keren gebruikt kan worden bij de definitie van een relatie. Major productnummer geeft het samenstellende deel weer van een product, bv fiets: major productnummer: bv fiets met productnummer 1, minor: componenten van bv een fiets, bv 2 wielen: major productnummer 1 en minor productnummer 2, quantity is 2 want 2 wielen in fiets. Je kan ook kijken naar een wiel en de spaken: major productnummer = 2 (== wiel) en minor productnummer 5 (spaken, waarvan je er bv 30 hebt), en dus is quantity 30. Eenzelfde domein kan dus meerdere keren gebruikt worden om attributen te karakteriseren. Attribuutwaarde == specifieke occurrence van een domein binnen een bepaalde tuple, geeft waarde van een attribuut weer. En elke waarde van een attribuut moet behoren tot het domein. Voor elk attribuuttype heeft een tuple precies 1 waarde. Die waarde kan ook null zijn: geeft weer dat de waarde onbekend of niet van toepassing is. Bv: student heeft geen (gekend) emailadres.

**Slide 10:** Belangrijk concept:

* Kandidaatsleutel == minimale determinant van een relatie en staat toe om op een unieke manier tuples te determineren/specifi¨eren binnen een relatie. Het kan ´e´en of meerdere attribuuttypes zijn waaruit die bestaat. Zelfde als in ER. Het moet altijd zo zijn dat de waarde van kandidaatsleutel kan gebruikt worden om unieke tuples in een relatie te gaan identificeren. Als het uit meerdere attribuuttypes bestaat (kandidaatsleutel), kan 1 attribuut dat onderdeel is van de kandidaatsleutel wel meerdere tuples gaan identificeren (dus niet uniek).
* Een relatie kan meerdere kandidaatsleutels hebben, we gaan 1 van die kandidaatsleutels promoveren als primaire sleutel → laat ons toe verbanden tussen relaties weer te geven. De anderen worden gedefini¨eerd als alternatieve sleutels.

Integriteitsconstraint: primaire sleutel kan nooit null zijn, moet altijd gekend zijn want zal gebruikt worden om verbanden tussen relaties weer te geven. Het moet altijd mogelijk zijn om op een unieke manier tuples binnen een relatie te identificeren.

**Slide 13:** Vreemde sleutel: laat u toe om verbanden weer te geven tussen relaties (zie ook definitie op de slide). In vb op slide: dnr is vreemde sleutel voor employee, is een verwijzing naar primaire sleutel dnr want laat u toe om het verband tussen employee en departement netjes weer te geven. Waarde van de vreemde sleutel is ofwel gelijk aan de waarde van een kandidaatsleutel in een ander tuple, ofwel null.

**Slide 14:** Vb: stel dat in de relatie departement een kolom wordt toegevoegd worden met verwijzing naar ssn (vreemde sleutel dus) die verwijst naar ssn in de relatie employee. Deze geeft alle werknemers weer die werken in een departement. Zou niet ok zijn want in een departement werken meerdere werknemers. Zou betekenen dat je allerlei ssn’s zou hebben die werken in het departement en zo cre¨eer je een meerwaardig attribuutnummer en dat mag niet. Je mag wel dnr als vreemde sleutel toevoegen aan employee want je kan maar in 1 departement tegelijk werken. Waarde van een vreemde sleutel moet ofwel met waarde in relatie departement ofwel met null overeenstemmen. Mag null zijn indien de minimumcardinaliteit 0 is. Indien minstens 1, mag die niet null zijn uiteraard.

**Slide 15:** Vb: relatie employee en project. Een employee kan aan meerdere projecten werken, dus je zou verschillende waarden moeten opnemen waaraan een werknemer gaat werken. Toevoegen van vreemde sleutel aan employee gaat hier niet. Aan project ook foreign key naar employee toevoegen: ssn. Maar aan een project kunnen meerdere werknemers werken, dus ook weer meerwaardig. We kunnen aan geen van beide relaties een foreign key toevoegen. Het probleem is dat de relatie m op n is ((m,n)). Nieuwe relatie introduceren (**Slide 16**): works on: waarbij je beide primaire sleutels van project en werknemer gaat combineren, beide vormen de primaire sleutel (de combinatie). De cardinaliteiten worden hier perfect ondersteund (werknemer kan aan meerdere projecten werken en projecten kunnen door meerdere werknemers uitgevoerd worden). Ook is het zo dat niet alle werknemers aan projecten moeten werken, dan zal die niet in works on verschijnen. De 4 cardinaliteiten (0..n) en (n..m). Heel belangrijk bij (n,m): altijd weergeven in het relationeel model met nieuwe tabel.

**Slide 17:** Een zeer belangrijke activiteit is het normaliseren. Is een stapsgewijs proces dat je gaat doorlopen om de redundantie weg te werken maar ook eventuele anomalie¨en en inconsistenties weg te werken.

**Slide 18:** Normaliseren: functional dependency. Bovenste tabel: heel wat overtolligheden: departementsnaam wordt meerdere keren opgeslagen, net als de locatie. Kan ook aanleiding geven tot tegenstrijdigheden wanneer men de gegevens gaat gebruiken. Stel dat een departement van naam verandert, dan moet je zoveel updates doorvoeren als er werknemers zijn die in een departement werken. Stel dat je de locatie verhuist, hetzelfde! → Niet effici¨ent. Wat als je een nieuw departement toevoegt? Dan ga je een nieuwe tuple aan de relatie toevoegen, maar daar werken nog geen werknemers in want het is nog in oprichting. Maar relatie employee heeft ssn als primaire sleutel, dus moet gekend zijn! Je gaat dan bv een ssn verzinnen om departementsgegevens toe te kunnen voegen → dummy waarden, is niet goed! Wat indien de laatste employee van een departement vertrekt → alle gegevens omtrent een departement kunnen verloren gaan. ⇒ Niet-genormaliseerde tabel (bovenaan) is dus niet netjes. Onderaan: 2 aparte tabellen en verwijzing naar elkaar. Is veel netter, consistenter en veel minder redunantie. Zowat alle problemen van zonet zijn weg. Hoe van de bovenste naar de onderste tabel? ⇒ Normalisatie!

**Slide 19:** Alle normalisatiestappen maken gebruik van functionele afhankelijkheden. Attribuut b is functioneel afhankelijk van attribuut a indien op elk moment in de tijd voor elke waarde van a er exact 1 waarde is voor b. Aangegeven door ”a → b”: betekent dus dat b functioneel afhankelijk is van a. Het omgekeerde is niet per se waar!

**Slide 21:** Eerste NF: geen samengestelde en meerwaardige attribuuttypen → waarde van attribuut moet atomair zijn en enkelvoudige waarde van dat domein. Eerste normalisatiestap bv: departement met dnr, dlocatie en dnaam met assumpties (waarmee je kan zien of in 1NF). Dnr is kandidaatsleutel (& primaire). Dlocatie is meerwaardig → relatie staat niet in 1NF. Dlocatie zal dus al niet functioneel afhankelijk zijn van de primaire sleutel. Oplossen: dlocatie uit relatie halen samen met primaire sleutel en in nieuwe locatie stoppen: dep location. Primaire sleutel van dep location: dnr EN dlocation (want dnr is ontoereikend want een departement kan meerdere locaties hebben).

**Slide 22:** Vb: departement R&D heeft 2 locaties, logistics ook. Dus splits op!

**Slide 23:** Ander vb: employee met ssn als primary key, ook ename en phonenr. Je weet niet of het in 1NF staat, tenzij je de assumpties kent! Assumpties tonen dat het niet in 1NF staat want meerdere telnrs zijn mogelijk! Meerwaardig attribuutnummer er weer uithalen en in nieuwe relatie steken: emp phone met phonenr en ssn. Ssn kan hier geen sleutel zijn want employee kan meerdere telefoons hebben. Phone number behoort tot 1 employee, dus toereikend als primaire sleutel. Stel dat dat gedeeld kan worden door meerdere employees: heeft impact op primaire sleutel! → Phone nr is niet langer toereikend, ook ssn zal nodig zijn.

**Slide 24:** Vb: niet in 1NF want employee kan aan meerdere projecten werken. **Slide 25:** 2NF: Prime-attribuuttype: maakt deel uit van kandidaatsleutel.

**Slide 26 & 27:** Relatie bovenaan: R 12 met SSN en PNR. Hebben overtolligheden: PNR InvestOptim wordt 2 keer opgeslagen, alsook Pduration → redunantie. Stel dat je de naam wilt aanpassen, zoveel veranderingen doorvoeren als er werknemers die aan het project werken. Pname is eigenlijk enkel functioneel afhankelijk van pnr. Duration is enkel functioneel afhankelijk van pnr. Hours is afhankelijk van de combo van ssn EN pnr!

Een relatie in 2NF begint met een relatie in 1NF en elk niet-primeattribuuttype (pnaam, pduration, hours) is volledig functioneel onafhankelijk van de kandidaatsleutel. Pnaam en pduration zijn enkel afhankelijk van pnr (volledig), dus apart zetten, want zijn niet afhankelijk van ssn.

2NF: issue wanneer kandidaatsleutel die uit 2 attribuuttypes bestaat, kijk of er geen non-primeattribuuttypes afhankelijk zijn van een stukje van de kandidaatsleutel. 2NF: non-primeattribuuttypen moeten *volledig* afhankelijk zijn van de kandidaatsleutel en niet van een deel van de kandidaatsleutels.

**Slide 28:** 3NF: relatie is in 3NF wanneer in 2NF en wanneer geen nonprimeattribuuttype transitief afhankelijk is van een kandidaatsleutel van de relatie. Transitieve afhankelijkheid: zie slide.

**Slide 29 & 30:** Relatie employee: SSN is kandidaatsleutel & primaire sleutel. Dnaam finance wordt 2 keer opgeslagen en managernummer ook! Weer overtolligheden dus!

Geen meerwaardige & samengestelde attributen, dus 1NF is ok. Elk nonprimeattribuuttype is volledig afhankelijk van kandidaatsleutel, dus ook in 2NF. Maar transitieve afhankelijkheid: dnummer is functioneel afhankelijk van ssn. Dnr gaat dname bepalen dus 2-staps functionele afhankelijkheid → van ssn naar dnr en van dnr naar dnaam dus dnaam is transitief afhankelijk van ssn via dnr. Hetzelfde met mgrnr: transitief afhankelijk van ssn via dnr. Naam en mgrnr eruithalen samen met het attribuuttype langs welk de transitieve afhankelijkheid werd vastgesteld.

**Slide 31-36:** Overgeslagen.

**Slide 37:** Tekortkomingen:

* Minimumcardinaliteit van 1 kan enkel afgedwongen worden indien de maximumcardinaliteit ook 1 is. Dus ”ten minste 1” bij aantal werknemers in department gaat niet zonder dat het maximum ook 1 is.
* Abstractie zoals bij (E)ER: specialisatie, generalisatie, disjointness en compleetheidsconstraints kunnen ook niet voorgesteld worden.

**Slide 38:** Vb: wat je in ER kan voorstellen, gaat niet in het relationeel model → geen perfecte afspiegeling/mapping!

**Slide 39:** Herhaling traject. Bemerk dat een ER-model niet perfect gemapt kan worden naar het relationeel model!

**Slide 40:** Vuistregels voor mapping:

* Voor elk entiteittype nieuwe relatie.
* Voor meerwaardig attribuuttype: nieuwe relatie die gekoppeld wordt via primair vreemde sleutelverband.
* ...

**Slide 41:** ER-model, in **Slide 42**: relationeel model ervan.

**Slide 43:**

* Kan een departement geen manager hebben? Nee: mag niet null zijn.
* Ja.
* Laatste 2: manager geen employee van zijn departement? & Can it be enforced? Zie volgende 2 slides!

**Slide 44:** Relationeel model gaat niet protesteren, maar eigenlijk heb je een situatie die je niet wenst. Is een tekortkoming van het ER-model dat ook in het relationeel model naar boven komt.

**Slide 45:** Kan niet afgedwongen worden!

**Slide 46:** Is het mogelijk? Dat kan en zal typisch het geval zijn recursief relationshiptype. Kan dus zijn dat je vreemde sleutel hebt die verwijst naar primaire sleutel/kandidaatsleutel van de vreemde relatie. Mogelijk dat employee door meer dan 1 employee gesupervised kan worden? Nee: eenwaardig. Met behulp van null-constraint minimumcardinaliteit afdwingen.

**Chapter 14**

# Huistaak vLes 12

## 14.1 Slides: Questions EER Normalisation StudentVersion(1)

**Vraag 1**

* A: specialisatie is parti¨eel bij a: is fout! Want t staat voor totaal!
* B: je gaat vertrekken vanuit een private client. Die kan 0..1 rekening hebben. Die rekening kan ofwel een savings account zijn, ofwel een current account, maar kan er dus maar 1 hebben en dus niet beiden tegelijk. Kan ze dus wel beiden hebben, maar niet gelijktijdig.
* **C is correct**: een account kan 1/meer klanten hebben voor 1 rekening en die klant kan zowel corporate als private zijn, dus correct!
* D: je hebt een current account, is een account (erft ervan). De eigenschappen van account zijn dat die 1/meer klanten kan hebben, dus je kan niet zeggen dat die maar 1 private client heeft. Je kan er meer hebben *en* die kunnen corporate of private zijn.

**Vraag 2**

* A is correct door de cardinality constraint: van bank statement naar account: maximaal 1.
* B: specialisation is inderdaad total en categorisation ook, dus **B is False**, dus het antwoord.
* C: is true want van private client naar account en dan naar bank statement: meerdere bank statements mogelijk, dus true.
* D: belong to multiple corporate clients want account kan tot multiple clients behoren, dus statement is true.

In het oorspronkelijk document op Toledo: B in de oplossing zegt A client entitiy (which is not a corporate or a private client) can have an account (which is not a savings or a current account). → nieuwere versie op Toledo!

**Vraag 3**

* A: een dish kan in meerdere restaurants gegeten worden, dus je weet niet wie wat waar heeft gegeten.
* B: is eigenlijk hetzelfde, maar dan in de andere richting: je kan vanuit het restaurant wel zien wat er is opgediend, van bepaalde schotels kan je weten wie dat heeft gegeten, maar niet waar ze het hebben gegeten.
* C: je ziet dat als je van restaurant naar dish gaat, dat nul is, je hoeft niks te serveren, je komt via dish uit bij persoon, waar ook nul staat, dus je kan een resto hebben zonder dat je een (indirecte) link hebt met een persoon.
* **D is correct**: als je een persoon hebt in het systeem, **moet** er ook een restaurant zitten in het systeem. Als je vanuit persoon vertrekt, moet die 1 schotel hebben die daarmee verbonden is en die moet verbonden zijn met minstens 1 resto hebt, dus als je een persoon hebt, heb je altijd een schotel en dus ook een restaurant.

**Vraag 4**

* A: cardinaliteit is ten minste 1.
* B: er staat niets dat het beperkt.
* C: minimumcardinaliteit bij restaurant vanaf dish is 1.
* **D is fout**: een persoon kan meerdere dishes eten bij meerdere restaurants, er staat niet at most one.

**Vraag 5**

* A: Overlappende specialisaties. Het is totaal, er zijn geen personenen die alleen maar persoon zijn (volgens deze zin), maar de p duidt op parti¨eel.
* B: head of gaat van persoon tot persoon, zet geen beperking op wie baas is van wie.
* **C is juist**: een persoon kan baas zijn van een andere persoon en die kan (per toeval) baas zijn van een manager.
* D: een persoon is een specialisatie van een manager: is fout. Persoon is generalisatie van manager.

**Vraag 6**

* A: ja, het is een EER model. Verschil: toegenomen semantiek bij EER.

Bij EER is overerving mogelijk, alsook generalisatie en categorisatie.

* B: p en o duiden erop.
* **C is fout**: de relatie zit tussen persoon en persoon. Deze kan clerk of manager of beiden tegelijk zijn. Clerk kan ook department head zijn van een manager. Er zijn geen cardinaliteiten aanwezig.
* D: clerk kan ook department head zijn van een manager want gaat van persoon tot persoon.

**Vraag 7**

* A: false: uitspraak legt het verband tussen gerechten die je eet en de reservaties die je maakt, maar die link staat niet in dat model. Er zijn gewoon 2 paden: van persoon naar restaurant, een via dish en de andere via reservation, maar paden zijn niet aan elkaar gelinkt.
* B: false: weer poging tot linking, maar geen pad ertussen. Als je vanuit persoon vertrekt, kan je niks zeggen over het aantal reservaties en het aantal gerechten. Zelfde vanuit restaurant. Je kan er dus geen uitspraken over doen.
* **C is true**: er wordt geprobeerd om ze met elkaar in verband te brengen. Als je gaat via restaurant, zie je dat als je een reservatie wilt hebben, je een restaurant *moet* hebben, maar geen dish. Als je via reservatie naar persoon gaat, moet je er minstens 1 hebben en via persoon naar dish is de minimum cardinaliteit ook 1, dus minimaal 1 dish!
* D: false: van persoon naar reservatie: het minimum is 0, die reservatie moet een restaurant hebben maar maakt niet uit want minimum bij reservatie is 0. Langs beneden: persoon *moet* een gerecht hebben en dat *moet* in een bepaald restaurant gemaakt zijn, dus het is verplicht voor een restaurant om een persoon te hebben.

**Vraag 8**

* Statement A is juist: we vertrekken van person en gaan naar dish, minimum cardinaliteit van 1, dus klopt.
* B: als je vertrekt van persoon via dish, minimaal 1, en van daaruit minimaal 1 restaurant.
* C: vertrek vanuit restaurant en kijk hoe deze gekoppeld is aan dishes. Er is geen verband tussen reservaties en dishes.
* D: reservaties en dishes zijn op geen enkele manier met elkaar verbonden, extra regels zouden moeten worden opgenomen, dus **D is false**.

**Vraag 9 B is juist!**: *C* : 3*x*4 = 12

*E* : 12*x*5 = 60

**Vraag 10** Functional dependency capteert constraints.

* **A is juist**.
* B: constraint is helemaal niet gecaptured.
* C: klopt ook niet want die combinatie is geen key dus die combinatie is niet gecapteerd.
* D: we zien dat een persoon, als de combinatie tussen persoon & company anders is, we dan niks kunnen zeggen over de date. RegID is de sleutel. Registraties gebeuren door een persoon voor een bedrijf op een bepaalde datum. Het enige wat je weet is dat als eenzelfde persoon zich bij hetzelfde bedrijf 2 keer registreert, dan *moet* dat op dezelfde datum. D zegt dat de persoon bij verschillende banken op verschillende data mag registreren.

**Vraag 11** Er is een transitieve functionele afhankelijkheid. In de relatie van bij email weet je de naam en het departement. Als er een transitieve functionele afhankelijkjheid is, staat het niet in 3NF. Als D juist is (ik denk dat ze C bedoelt), is A dus ook juist, dus geen van die 2. De relatie is in 2 NF (samengestelde sleutel: alles is afhankelijk van de volledige sleutel en niet van een stuk ervan). De relatie is in 2NF klopt dus.

D slaagt op niks want iets is parti¨eel afhankelijk van de sleutel of volledig afhankelijk van de sleutel. **D is fout**.

**Vraag 12**

* Als iets in 3NF staat, dan ook in 2NF en 1NF. A en B kunnen dus al niet.
* We weten dat C verboden is.
* D is definitie van 2NF en aangezien het in 3NF staat, dus ook in 2NF, dus **D is correct**.

**Vraag 13** Professorveld is niet atomisch, dus niet genormaliseerd. **A is correct**.

**Vraag 14** Je hebt afhankelijkheden. 2NF gaat over een parti¨ele afhankelijkheid van de sleutel, dus 2NF is alleen relevant indien je een samengestelde sleutel hebt, wat je hebt. Dan heb je de afhankelijkheid restaurant bepaalt name, dus met andere woorden, die name is niet afhankelijk van de volledige sleutel, maar van een stuk van de sleutel, nl restaurant. Je weet dat deze relatie niet in 2NF staat. De uitspraak ”is in 2NF” is dus fout. 3NF dus ook. De relatie is in 1NF is juist want er zijn geen multivalued attributes. **B is dus juist**. Gedeeltelijk genormaliseerd want in 1NF en dus niet volledig genormaliseerd. Je weet dat als 3NF waar is, dan ook 1NF en 2NF, kan dus niet want dan is er meer dan 1 antwoord juist. Je weet dus dat je moet kiezen tussen A en B.

15 examenvragen van Professor Snoeck. Minstens 2 vragen over ER en minstens 1 `a 2 over normalisatie.

**Chapter 15**

# Les 13: 23/03/2015

## 15.1 Slides: 4D SQL

**Slide 6:** We hebben de informatie gemodelleerd aan de hand van modellen en nadien moeten we die kunnen populeren met informatie. Als we het hebben over relationele modellen, gaan we die gegevens ophalen door middel van SQL.

**Slide 7:** SQL dient voor het opvragen van gegevens, maar ook om tabellen aan te maken. We kunnen data defini¨eren (data definition language), maar ook opvragen en manipuleren.

**Slide 8:** Voorbeeld van SQL-code waarin nieuwe relatie wordt gemaakt. We maken een onderscheid tussen alfanumerieke gegevens (char, kan/moet je niet mee rekenen. Postcode: je gaat er niet mee rekenen) en numerieke. Beschrijving per kolom is mogelijk.

**Slide 9:** Data manipuleren en opvragen. 4 vbn: data ophalen: eerste query gaat namen en telefoonnummers ophalen van personen, tweede is om data toe te voegen, derde: gegevens bewerken, vierde: gegevens verwijderen. Als we verwijderen, gebruiken we meestal de primary key om zeker te zijn dat het verwijderd is.

**Slide 10:** SQL is set-geori¨enteerd en declaratief. Er is een standaardtaal, maar er zijn heel veel varianten.

* SQL maakt gebruik van het relationeel model: opereert op verzamelingen en geeft resultaten terug in termen van verzamelingen.
* Zo komen we bij de tweede belangrijke eigenschap: is declaratief: geeft aan welke data moet worden opgehaald, niet hoe dat moet gebeuren. Is heel belangrijk want we gaan dus declaratief programmeren en niet procdureel.
* Het is belangrijk te weten dat er verschillende dialecten zijn. SQL is gestandardiseerd, maar dat is veel te laat gebeurd.

**Slide 11:** SQL kan op 2 manieren gebruikt worden: SQL-statements invoeren in de terminal en dan wordt dat interactieve SQL. We kunnen het ook embedden in de broncode in de source code van een programma.

**Slide 12:** Access

**Slide 13:** Eclipse: SQL ingebed in Java-code.

**Slide 14:** We gaan het hier enkel hebben over het consulteren van data en niet over het manipuleren: SELECT en enkel in interactieve context gebruiken. In embedded context ga je bijna dezelfde statements tegenkomen, dus niet zoveel verschil.

**Slide 17:** SQL SELECT wordt gebruikt om gegevens op te halen. Belangrijk op te merken dat de SELECT-operatie nooit gegevens zal verwijderen of wijzigen. Kan wel gegevens anders weergeven dan ze in de databank staan, maar ze zal nooit gegevens wijzigen of verwijderen. Veranderingen zullen dus nooit permanent zijn of gegevens veranderen voor andere gebruikers.

**Slide 18:** SELECT-operatie kan uit verschillende componenten bestaan. Daarnaast heeft SELECT als voornaamste doel om gegevens te projecteren. SELECT geeft aan welke kolommen van de tabel weergegeven moeten worden. We gaan dus een subset weergeven.

**Slide 19:** SELECT kan uit verschillende componenten bestaan: SELECT geeft aan welke kolommen we willen selecteren. De enige verplichte component is FROM: we willen aangeven over welke tabellen we het hebben. SELECT geeft aan welke kolommen je wil projecteren en FROM geeeft aan vanuit welke tabel dat moet.

* WHERE: conditie op de data opleggen: welke deelverzameling van de data willen we tonen?
* GROUP BY: gegevens in groepjes samenvoegen.
* HAVING: legt de conditie op op de gegroepeerde data. v WHERE op de gewone data.
* ORDER BY: gaat aangeven hoe we de gegevens willen sorteren.

**Slide 20:** SELECT uitgevoerd door computer: belangrijk bij het begrijpen van SQL. We weten dat de query begrepen gaat moeten worden door de computer. Als wij de engine zouden zijn, willen we eerst weten waar het over gaat, dus:

1. FROM-component: deze gaat zeggen welke tabellen we willen opvragen.
2. WHERE-component wordt beoordeeld want deze legt een conditie op de ongegroepeerde data en zo zie je welke data wordt overgehouden.
3. GROUP BY
4. HAVING: om een extra conditie op de gegroepeerde data te leggen.
5. SELECT en DISTINCT: gaat zeggen welke kolommen van de opgevraagde gegevens we willen projecteren.
6. ORDER BY: projectie sorteren.

**Slide 21:** Wat wij vragen wordt dus helemaal anders ge¨ınterpreteerd door de PC. Pas op het allerlaatste moment gaat die SELECT uitvoeren. Die volgorde impliceert niet dat SQL procedureel is! Geeft gewoon aan hoe de PC de gegevens ophaalt.

**Slide 24:** FROM-component: aangeven welke tabellen we willen bevragen. In het geval van meerdere tabellen moeten we aangeven wat de relatie van die 2 tabellen tot elkaar is. 1 tabel is natuurlijk veel makkelijker dan 2/meer.

**Slide 25:** Bevraging van 1 tabel ligt voor de hand: in FROM maar 1 tabel meegeven: we selecteren alle vakken en projecteren enkel de code en de titel daarvan.

**Slide 26:** In de realiteit gaan we vaak meerdere tabellen tegelijk bevragen omdat we bij het defini¨eren van de databank onze gegevens hebben moeten normaliseren. Bij de normaalvormen moeten we de tabellen opsplitsen, dus nadien zullen die tabellen weer met elkaar gelinkt moeten worden. Als we de foreign key en de primary key gaan ophalen, zien we hier op slide 26 het vb: 1 prof kan meerdere vakken doceren, maar omgekeerd niet. We gaan 1 prof met meerdere vakken linken, dus we willen de tabellen achteraf weer aan elkaar linken door middel van de foreign en de primary key.

**Slide 27:** Als we de FROM-component uitvoeren, doen we dat in 2 stappen. We voeren eerst het carthesisch product uit. Als tweede stap gaan we een conditie op de data leggen, via een JOIN- of WHERE-conditie. Zie voorbeeld **Slide 28 & 29**.

**Slide 28:** Stap 1: carthesisch product: alle data met elkaar combineren, is duidelijk niet hetgene we willen want er staat foute data in. We willen enkel de records overhouden die matchen met elkaar.

**Slide 29:** ProfID uit de ene tabel moet overeenkomen met het ID uit de andere tabel: foreign key uit de ene tabel moet overeenkomen met de primary key uit de andere tabel. In de tweede stap willen we dus alle nietszeggende gegevens verwijderen in het resultaat (*niet* in de databank!). We krijgen in dit geval zowel bij JOIN als bij WHERE hetzelfde resultaat. De prof verkiest een inner join omdat het aangeeft waartoe de conditie dient.

**Slide 31:** Verschillende soorten JOIN. Soms beter dan WHERE omdat er verschillende soorten JOINs zijn. Heel goed aan te geven via verzamelingenleer. Bv: INNER JOIN: doorsnede van 2 tabellen.

**Slide 32:** Resultaat van een INNER JOIN is de intersectie van 2 verzamelingen.

**Slide 33:** Enkel de records aan elkaar koppelen waarbij de IDs matchen. NULL: waarde in een record is onbekend. Hier interpreteren als: de prof van macro-economie is onbekend *of* die heeft geen prof. We gaan dus enkel de records opnemen die met elkaar matchen.

**Slide 34:** LEFT OUTER JOIN: gaat rijen van de linkerkant weergeven: geeft alle rijen van de linkerkant weer en gaat dan kijken naar de rechterkant; als daar geen match is: NULL. Het gaat voor alle duidelijkheid over links en rechts in de query, niet de plaats van de tabellen.

**Slide 35:** Linkerkant query: Professor, allemaal opnemen en dan zoeken naar een match in de rechterkant: alle professoren worden weergegeven. Als er geen match is: NULL-waarden.

**Slide 36:** RIGHT OUTER JOIN doet exact hetzelfde als de LEFT, alleen omgekeerd.

**Slide 37:** Vb: rechterkant is Course: alle records uit de Course-tabel opnemen en dan zoeken naar de Professor-waarden. Indien geen prof gevonden: NULL.

**Slide 38:** FULL OUTER JOIN: gaat LEFT en RIGHT combineren. Indien geen match: NULL. Dan links EN rechts NULL (zie **Slide 39**).

**Slide 40:** Alias: andere naam geven aan een tabel/kolom. Kan in FROM en SELECT gebruikt worden. Alias verwijdert/verandert dus ook niets in de tabel! Is gewoon in de query dat je het aanpast. Vb: Person AS p en dan voor zijn elementen ”p.” zetten. In dit vb is dit niet nodig (eerste), maar in tweede wel want daar gebruik je dezelfde tabel 2 keer en je wilt die 2 keer anders gebruiken. Eerste keer personentabel als a en daarna als b en nadien de tabel joinen op basis van id, terug gebruiken in SELECT.

In derde vb: alias voor kolom: alle records uit persoon en dan naam en telnr projecteren. Telnr gaan we anders noemen in de uitvoer, nl Number. Enkel de gebruiker die de query gaat uitvoeren/gebruiken, zal de andere naam zien. Naam zal niet worden aangepast in de tabel zelf.

**Slide 41:** Gewoon het carthesisch product: 4 records: 2 records in de ene en 2 in de andere.

**Slide 44:** WHERE gaat een conditie op data leggen: data verkregen uit FROM. We weten nu hoe de SQL-engine werkt: eerst FROM (welke tabellen willen we?) en nadien die data filteren door de WHERE conditie.

**Slide 45:** SELECT gaat projecteren en WHERE gaat selecteren. We gaan dus alle records selecteren die voldoen aan de conditie.

**Slide 46:** We kunnen de conditie in de WHERE-component samenvoegen met behulp van bouwstenen: operators om gegevens met elkaar te vergelijken. Zie tabel. Die operators kunnen toegepast worden op numerieke en alfanumerieke data. Bij numerieke data: geen aanhalingstekens, bij alfanumerieke data: aanhalingstekens. Eerst worden de klanten geselecteerd waarbij het klantennummer gelijk is aan 2, nadien worden alle kolommen van de gegevens die eraan voldoen geprojecteerd. Hetzelfde met alfanumerieke data en de naam.

**Slide 47:** Er zijn ook booleaanse operatoren: wordt gebruikt om condities samen te voegen. In dit vb gaan we eerst de klanten selecteren die in Germany of Mexico wonen en daarna de klanten die daaraan voldoen projecteren. Zien we door het sterretje in de SELECT-component.

**Slide 48:** IN laat toe om de elementen in 2 verzamelingen met elkaar te vergelijken. SQL geeft resultaten weer in verzamelingen. Zal in dit geval hetzelfde doen als in dezelfde operator.

**Slide 49:** BETWEEN: bereik van verschillende waarden opleggen.

**Slide 50:** Operator die enkel op alfanumerieke data kan gebruikt worden. Gaat die testen op bepaalde patronen. Is niet hetzelfde als de vergelijkingsoperator; je gaat testen op patronen en niet op waarden. Je kan hier wildcards gebruiken, waar de % op slaagt. In dialecten kan de wildcard ook een \* zijn. Belangrijk om op te merken dat het symbool voor wildcard afhangt van het dialect.

**Slide 51:** NULL geeft aan dat de waarde onbekend is. Gaan anders behandeld worden: speciale operator voor nodig: IS NULL of IS NOT NULL. Want het is iets helemaal anders dan gewone data.

**Slide 52:** Selectie maken uit programme-tabel en enkel records selecteren waarvoor Level NULL is. En in tweede vb net niet.

**Slide 52:** Like: enkel voor alfanumeriek. Waarom geen ’=’ voor alfanumeriek? Like wordt gebruikt als pattern matching. Verschil zit hem in het gebruik van wildcards.

**Slide 56:** Query met wildcard (\*) gaat alle kolommen van een bepaalde tabel projecteren. In tweede vb: enkel kolommen van de orders-tabel projecteren. Eerst over 2 tabellen gaan queryen en nadien enkel de resultaten van ´e´e ervan tonen.

**Slide 57:** Verschil tussen productid en kwantiteit ”result” noemen en die projecteren. Tweede vb: gaat gewoon 5 tonen als Five (dus letterlijk een kolom met allemaal Five-en).

**Slide 58:** Query uitvoeren waarvan we weten dat er dubbels gaan zijn, daar gaan we DISTINCT voor zetten. Elke verzameling die we teruggeven als resultaat, daarin moet elk element uniek zijn. Gaat dus enkel de dubbels uit het resultaat halen en niet verwijderen/aanpassen want we werken met SELECT. In het vb krijgen we een lijst van unieke landen. SELECT DISTINCT is niet hezelfde als GROUP BY. Bij die laatste ga je de kolommen groeperen aan de hand van de waarde ervan. Bij GROUP BY blijven er waarden over, DISTINCT gaat dubbels uit resultaat verwijderen.

**Slide 59:** Gaat ´e´en enkele waarde berekenen op basis van verschillende waarden == static functions,... (zie slide).

**Slide 60:** count(\*) != count([kolomnaam]). (\*) gaat alle rijen tellen inclusief NULL-waarden. In tweede vb: zonder NULL. In derde vb: gemiddelde berekenen.

**Slide 61:** We gaan met FROM de twee records in de tabel selecteren en voor elke record daarvan gaan we 5 + 2 doen. Per aggregaatfunctie gaat de som daarvan genomen worden (dus 7 + 7) en dan krijgen we dus 14 terug. De kolomnaam gaat ’unknown’ heten.

**Slide 64:** ORDER BY wordt pas op het einde uitgevoerd en gaat dus sorteren. Zegt hoe kolommen in het resultaat moeten gesorteerd worden.

**Slide 65:** Ascending: oplopend, descending: aflopend. Country tabel moet oplopend gesorteerd worden en de postcode moet aflopend gesorteerd worden.

**Slide 68:** Doel is om de waarden te groeperen van bepaalde kolommen. Is niet hetzelfde als SELECT DISTINCT, die gaat dubbels vewijderen. GROUP BY gaat waarden groeperen.

**Slide 69:** In dit vb zal het duidelijk worden wat bedoeld wordt met ’groepeer waarden van kolommen’: orders nemen en GROUP BY ShipperID. In tweede tabel (rechtsonder) zie je het resultaat. Orders zijn samengevoegd bij ShipperID. Resultaat van de groepering is dat de 3’s, de 2’s en de 1’s worden samengevoegd. Als je dan alleen ShipperID gaat projecteren, krijg je dus enkel 1, 2, 3. je moet je voorstellen dat de andere gegevens (OrderID,...) daar nog achter zitten. We kunnen nog tellen hoeveel 3’s er waren en het gemiddelde berekenen van de EmployeeID (om welke reden dan ook).

**Slide 70:** Orders nemen en sorteren op shipperID en dan gaan we de ShipperID projecteren en het aantal projecteren: we kunnen nog altijd tellen hoeveel records een 1 hadden bv. Is het verschil tussen GROUP BY en DISTINCT. Kan ook met andere functies dan count gebruikt worden (ook avg,...).

**Slide 71:** GROUP BY gaat de achterliggende gegevens behouden, distinct gaat dubbele waarden weggooien in de projectie. Dat verschil wordt pas echt duidelijk wanneer je naar de stappen van de SQL-uitvoering gaat kijken. GROUP BY gebeurt voor SELECT en DISTINCT erna. Als je groepeert, behoud je dus nog de data over de groepjes, dus kan je nog aggregaties toepassen.

**Slide 74:** Doel van HAVING: wordt vlak na GROUP BY uitgevoerd. We hebben dus eerst WHERE, dan GROUP BY en dan HAVING. WHERE legt een conditie op de ongegroepeerde data, dan groeperen en dan HAVING op gegroepeerde data. HAVING gaat een conditie leggen op de geaggregeerde data waardoor het mogelijk is om de uitkomsten van die geaggregeerde functies te testen. Is dus niet hetzelfde als WHERE, wat je kan dat zien door de volgorde van uitvoering. Bij WHERE kan je niet testen want er is nog geen groepering.

**Slide 76:** In eerste vb order selecteren van werknemer met ID == 6. Daarna gaan we de data groeperen op basis van de ShipperID’s. Eerst doen we de WHERE, dan groeperen we de data op basis van ShipperID. In het tweede vb doe je hetzelfde en op het einde tel je het aantal Orders. HAVING zegt dat je enkel de Orders gaat projecteren waarbij de count hoger is dan vier. Dus shippers die op z’n minst 4 orders hebben afgehandeld.

**Slide 78:** Weblectures: subqueries etc. Subqueries worden opnieuw behandeld als een verzameling (hun uitkomst). In SQL kan je een SELECT zien als een verzameling. Dus we kunnen een subquery opnemen in de FROM en in de WHERE. Wanneer in de FROM: zal behandeld worden als het equivalent van een tabel. Als we een subquery gaan toevoegen in de WHERE-conditie hebben we 2 gevallen: een die 1 enkele waarde/query gaat teruggeven, maar ook een die meerdere records gaat teruggeven. Als subquery in de WHERE 1 waarde teruggeven, kan je ”=” gebruiken, anders kan je IN gebruiken en andere operatoren die verzamelingen met elkaar vergelijken.

Set operaties: 2 queries aan elkaar koppelen: unie, doorsnede,...

**Chapter 16**

# Huistaak vLes 14

## 16.1 Slides: 4C SQL

### 16.1.1 Video: Business Information Systems 4: Structured Query Language (SQL)

Slides zijn niet in dezelfde volgorde als in de video. Een mapping is gemaakt.

**Slide 1:** We gaan aan genormaliseerde databanken vragen stellen met behulp van SQL.

**Slide 8:** SQL is een database language die je kan gebruiken om gegevensstructuren te defini¨eren: Data Definition Language → set van instructies die je toelaten om tabellen te defini¨eren, primaire sleutels, attribuuttypen en vreemde sleutels. Naast DDL kun je SQL ook gebruiken voor DML: Data Manipulation Language: data ophalen, invoegen, verwijderen en updaten.

SQL is een set-ge¨orienteerde taal en gaat dus met verzamelingen werken. Enerzijds is een verzameling een set van ongeorganiseerde tuples, het resultaat van SQL is op zich ook een verzameling, die dan kan gebruikt worden voor allerlei andere doeleinden. Het resultaat van en SQL statement is typisch een verzameling of terug een tabel van resultaat-tuples.

SQL is zeer een geavanceerde taal want je moet de queries niet manueel gaan programmeren. Vroeger moest men procedureel de queries programmeren. Bij SQL is dat niet langer het geval, zeer gebruiksvriendelijk. Het heeft een zeer sterke, mathematische onderbouw. Het is gebaseerd op relationele algebra en relationele calculus. SQL kun je op verschillende manier gebruiken: interactief (via toepassing waarbij je je query kan intypen, al dan niet manueel → eindgebruiker gaat interactief queries formuleren al dan niet aan de hand van een front-end programma of via natural programming) en ingebed (zie volgende slide).

**Slide 9:** Access: zelf je query intikken met alle tabellen en condities volgens dewelke de gegevens moeten opgehaald worden. Naast het interactief uitvoeren, kun je SQL ook ingebed in een programmeertaal uitvoeren.

**Slide 10:** We hebben supplier, purchase order en product met elk attribuuttypen.

**Slide 11:** Hoe gaan we dat mappen naar een relationeel model? Zie slide. We hadden tussen supplier en product een (n,n)-relationship type. En (n,n) geeft altijd aanleiding tot een extra tabel: Supplies.

**Slide 12:** We gaan dit implementeren via SQL DDL.

**Slide 13:** Verdere definitie.

**Slide 14:** Verdere definitie.

**Slide 15:** Verdere definitie.

**Slide 19:** (Titel komt niet overeen, maar is zelfde slide) On delete/update cascade/restrict: je hebt een suppliertabel en suppliestabel. In die laatste heb je supnr en prodnr die samen de primary key vormen. Stel dat je in supnr Jenkins zou verwijderen, dan zit je bij supplies met een probleem want je hebt 2 tuples met supnr 32. Aan de hand van on delete cascade: ook de verwijzingen naar Jenkins worden verwijderd. Bij on delete restrict ga je ervoor zorgen dat wanneer je wilt verwijderen en er is nog een verwijzing, dat niet mag. On update cascade/restrict werkt analoog.

**Slide 17:** We gaan kijken naar eenvoudige queries: geformuleerd in SQL die data van slechts 1 tabel gaat ophalen.

**Slide 18C:** Eerste vb zegt dat je alles wilt van de tabel supplier. Je kan daar ook een shortcut notatie voor gebruiken: in plaats van alles op te sommen, gewoon \* gebruiken.

Als je in slechts enkele kolommen ge¨ınteresseerd bent, geef dan enkel die kolommen mee. Als je geen dubbels wilt in de ordening/opvraging, gebruik DISTINCT.

**Slide 19:** Voorbeeld van tabel.

**Slide 20:** Ook een tabel, hoort bij die van **Slide 19C**.

**Slide 21:** Die simpele queries verder complex maken door in de WHERE component een verfijning op te leggen aan de data select, zie slide.

**Slide 22:** Duidende voorbeeldjes.

**Slide 25:** Ook aggregaatfuncties zoals COUNT, MIN, MAX, SUM, AVG, STDEV, VARIANCE mogelijk.

**Slide 26:** Voorbeeldjes.

**Slide 27:** Tabellen om vbn op toe te passen op volgende slide.

**Slide 28:** Tabel toegepast op queries **Slide 19**.

**Slide 29:** Verdere voorbeelden: average, standaarddeviation, variantie, minimum en maximum. Slide in video verschilt lichtjes van de gegeven slides, essentie staat er wel op.

**Slide 30:** Gesoficisticeerder: GROUP BY in combinatie met HAVING: met GROUP BY kunnen we groepjes maken van tuples die bepaalde karakteristieken hebben. De HAVING constructie zal dan toelaten om een selectie van die groepjes te verfijnen. **HAVING kan enkel voorkomen in combinatie met**

**GROUP BY.**

**Slide 31:** Ook bij HAVING zijn aggregaten mogelijk. In het eerste vb: binnen purchase order model: verschillende groepjes. We maken groepjes voor prod nr 117, 118 en dan kijken we binnen elk van die groepjes of er strikt meer dan 1 tuple aanwezig is. Indien dat zo is, wordt het productnummer gerapporteerd. Vb 2: we kunnen ook kijken naar het ophalen van de productnrs, de som nemen van de quantity gegroeperd bij prod nr waarbij de som van de kwantiteit in een groepje groter is dan 100.

**Slide 32:** Tabel om de vorige slide op toe te passen. GROUP BY prod nr gaat kijken naar de tabel en verschillende groepjes maken, zoals op **Slide 26**.

**Slide 16.1:** (Deze slide hebben we niet, zie Figuur 16.1) We krijgen de volgende 5 groepjes, geordend op prod nr. Dit is wat GROUP BY gaat doen. De eerste query ging kijken of er binnen die groepjes meer dan 1 tuple aanwezig is, voor 118 is dat waar, voor 1089 is het niet waar, want maar 1 tuple. Analoog bij de andere groepjes. De tweede query ging kijken naar HAVING SUM groter dan 100, voor 118 klopt dat en voor 1089 ook, voor de rest niet. We hebben maar 2 groepjes waar de som groter is dan 100.

**Slide 33:** Je kunt nu heel mooi zien hoe die 2 vorige queries ge¨evalueerd worden. Resultaten zijn te zien op deze slide.

**Slide 23:** ORDER BY: resultaten van de query sorteren, eender welke kolom. Default is altijd oplopend. Interpretatie van NULL waarden: moet je checken in uw database software pakket, hangt daar vanaf! Dat verschilt naargelang de implementatie en de databasesoftware die je gaat gebruiken.

**Slide 24:** Voorbeeld: sorteren op basis van datum, oplopend en supnr descending. Voorbeeld 2 met order by 3: dan ga je ordenen volgens het derde attribuuttype in het select statement, in dit geval de price.

**Slide 34:** Gegevens bevragen van meerdere tabellen: JOIN.

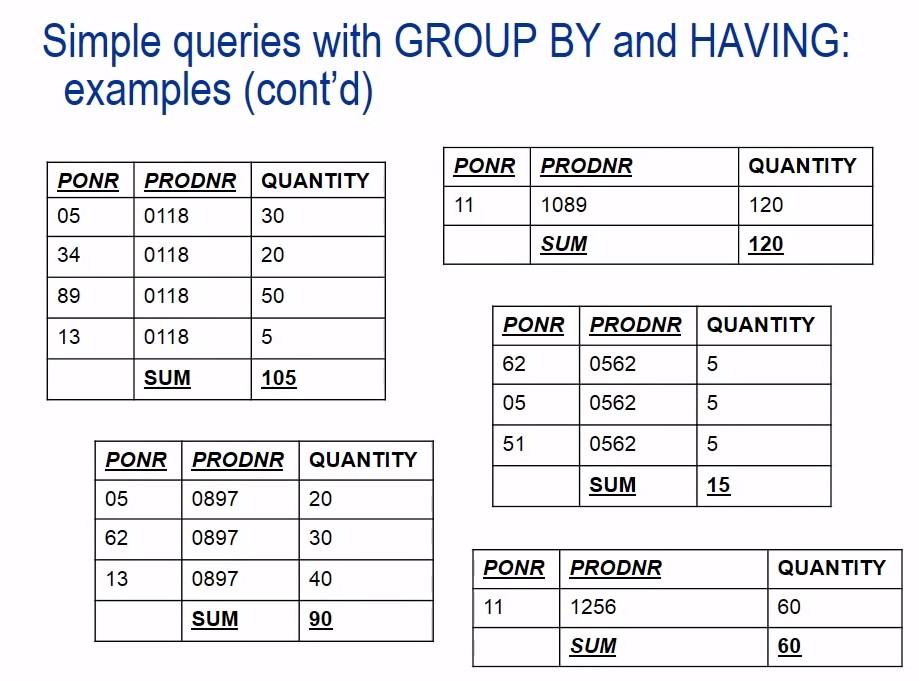


Figure 16.1: Ontbrekende slide.

**Slide 36:** Vb: als je naar de query kijkt, is het eerste wat je altijd moet doen, de tabellen oplijsten. We hebben dus gegevens nodig uit supplier en supplies. Voor de supplier wordt een shortcutnotatie r gebruikt, voor supplies s. De database zal het carthesisch product nemen van beiden: tabel in memory die elke combinatie van supplier en supplies gaat maken. Dat betekent dat er heel wat suppliers gaan gekoppeld worden aan foute supplies, maar je wil dat de koppelingen correct gelegd worden. Vandaar in de WHERE clause de bijkomende restrictie: supplier nummers moeten overeenkomen. Zo kan je er zeker van zijn dat de gegevens correct gecombineerd worden. → JOIN: gegevens van meerdere tabellen samenvoegen. Tweede query doet hetzelfde als de eerste, gewoon met JOIN.

**Slide 37:** Tabel bij voorbeeld.

**Slide 38:** Resultaat van de JOIN met de tabellen van vorige slide.

**Slide 39:** Tweede voorbeeld. Gegevens nodig uit 4 tabellen die aan elkaar gelinkt zijn met primaire vreemde sleutelverbanden.

**Slide 40:** Query bij vb 2.

**Slide 41:** Resultaattabel van vb 2.

**Slide 42C:** Je kan ook gegevens JOINen met zichzelf. We zoeken naar alle paren van leveranciers die in dezelfde stad gevestigd zijn. We hebben dus gegevens nodig uit de suppliertabel. De query ziet er dan als volgt uit:

SELECT R1. SUPNAME, R2.SUPNAME, R1.SUPCITY

FROM SUPPLIER R1, SUPPLIER R2

WHERE R1.SUPCITY = R2.SUPCITY

AND (R1.SUPNR < R2.SUPNR);

We nemen dus de suppliertabel en nemen 2 verschijningsvormen. Daarna combineer je ze, enkel wanneer de suppliercity overeenkomt. Waarom doen we R1.SUPNR *<* R2.SUPNR? Illustreren aan de hand van het voorbeeld op de volgende slide.

**Slide 43:** We hebben de 2 verschijningsvormen van de suppliertabel. De tabellen zijn identiek. We gaan die matchen en welke suppliers wonen in dezelfde gemeente? Jenkins en Quinn. Stel nu dat je die laatste conditie weglaat (R1.SUPNR *<* R2.SUPNR). Wat je dan krijgt is Jenkins en Jenkins want ze wonen beide in Antwerpen, Abbott en Abbott, Nelson en Nelson en Quinn en Quinn, maar dus ook Jenkins en Quinn. Die laatste conditie wordt dus toegevoegd om ervoor te zorgen dat je die dubbels eruit kunt halen. Je had het ook omgekeerd kunnen doen, dus: R1.SUPNR *>* R2.SUPNR, dan had je Quinn en Jenkins als resultaat (dus omgekeerd).

**Slide 44:** We kunnen naast de JOIN-condities ook andere zaken voor verfijning opleggen, bv selecteer de namen van alle suppliers die het product met productnummer 1107 kunnen supplyen. De bijhorende query is dus:

SELECT R.SUPNAME

FROM SUPPLIER R, SUPPLIES S

WHERE R.SUPNR = S.SUPNR

AND S.PRODNR = ’1107’;

**Slide 45:** Tabellen voor deze query met bijhorend resultaat.

**Slide 46:** Ander vb: minder belangrijk. Soms kunnen er in het resultaat ook duplicaten aanwezig zijn, die kan je eruit filteren met DISTINCT:

SELECT DISTINCT R.SUPNAME

FROM SUPPLIER R, SUPPLIES S, PRODUCT P

WHERE S.SUPNR = R.SUPNR

AND (S.PRODNR = P.PRODNR)

AND (P.PRODCOLOR = ’GREEN’);

Die DISTINCT zorgt ervoor dat indien er suppliers zijn die meerdere groene producten kunnen leveren, die slechts 1 keer gerapporteerd worden. Je zou het kunnen linken aan een van de eerdere opmerkingen en zeggen dat het resultaat van een SELECT-instructie is altijd een set, dus kunnen we eventueel die DISTINCT weglaten, dat zou een terechte opmerking zijn. Dus voor dit voorbeeld is DISTINCT niet relevant.

**Slide 47:** Find product number, name and total ordered quantity for each product that is specified in an order. Als je een complexe query ziet, lees deze 2 keer.

SELECT P.PRODNR, P.PRODNAME,

SUM(POL.QUALITY)

FROM PRODUCT P, PO\\_LINE POL

WHERE P.PRODNR = POL.PRODNR

GROUP BY P.PRODNR;

Deze query gaat de producttabel naast de purchase order line (PO LINE) tabel zetten en mergen op basis van het overeenstemmend productnummer. Je gaat de samengevoegde tabellen bekijken, daarbinnen groepjes aanmaken en per groepje het productnummer rapporteren, productnaam en de som van de hoeveelheid. **Slide 48:** Tabel met toepassing vorige query.

**Slide 49:** De net besproken JOINs zijn INNER JOINs, je hebt ook OUTER JOINs: je gaat altijd proberen om een koppeling te leggen. Indien dat niet kan, ga je aanvullen met NULL-waarden.

**Slide 50:** Retrieve number, name and status of all suppliers and if applicable, include number and price of the products they can supply. Belangrijk hier is dat je *alle* leveranciers nodig hebt. Een outer join kan je herkennen aan het woord ’alle’ en hier heb je dus alle leveranciers nodig. Query:

SELECT R.SUPNR, R.SUPNAME, R.SUPSTATUS, S.PRODNR, S.PURCHASE\_PRICE

FROM SUPPLIER AS R LEFT OUTER JOIN

SUPPLIES AS S

ON (R.SUPNR = S.SUPNR);

Als je een supplier hebt die geen supplies kan leveren, geef je die leverancier ook weer, maar met NULL-waarden. Dat kan je realiseren met een LEFT OUTER JOIN: je wilt alle tuples in de linkertabel (hier supplier) mee in het eindresultaat. Als je een koppeling kan maken, leg je die, anders vul je aan met NULL-waarden.

**Slide 51:** Tabellen bij voorbeeld.

**Slide 52:** Resultaat van de query op de voorbeeldtabellen.

**Slide 53:** Ander voorbeeld: select all product numbers, together with their product name and total ordered quantity, even if there are no outstanding orders for a product at the moment. Query:

SELECT P.PRODNR, P.PRODNAME,

SUM(POL.QUANTITY) AS SUM

FROM PRODUCT AS P LEFT OUTER JOIN PO\_LINE AS POL

ON (P.PRODNR = POL.PRODNR) GROUP BY P.PROD\_NR;

**Slide 54:** Tabellen bij voorbeeld.

**Slide 55:** We kunnen dan gaan kijken naar geneste queries: queries binnen queries. Je hebt altijd een outer block en een inner block, dus een binnenste blok. Het systeem zal altijd hetzelfde toepassen: eerst binnenste blok oplossen en dan naar het buitenste blok gaan.

**Slide 56:** Retrieve the name of the supplier where a purchase order with a specific number is placed. Je kan dit met een JOIN uitvoeren, maar ook met een geneste query. Query:

SELECT SUPNAME

FROM SUPPLIER

WHERE SUPNR =

(SELECT SUPNR

FROM PURCHASE\_ORDER

WHERE PONR = ’79’);

Het systeem gaat altijd eerst naar het inner block gaan. Je krijgt daarvan bepaalde suppliernummers terug. Het is een scalaire subquery omdat de inner block een bepaalde waarde oplevert, een waarde die we dan ook een scalar noemen.

**Slide 57:** Ander vb: for each product that exceeds the available quantity of a specific product, retrieve the number and name. Kan met JOIN maar ook met scalaire query:

SELECT PRODNR, PRODNAME

FROM PRODUCT

WHERE AVAILABLE\_QUANTITY >

(SELECT AVAILABLE\_QUANTITY

FROM PRODUCT

WHERE PRODNR = ’0117’);

We hebben weer een outer en inner block. Systeem gaat dus weer het inner block eerst bekijken; stel dat het systeem 100 teruggeeft, dan krijg je AVAIL-

ABLE QUANTITY *>* 100

**Slide 58:** Tabel subquery: tabel als resultaat, niet 1 getal. Bv: retrieve all supplier names who can supply a specific product. Query:

SELECT SUPNAME FROM SUPPLIER

WHERE SUPNR IN

(SELECT SUPNR

FROM SUPPLIES

WHERE PRODNR = ’0117’);

Binnenste blok gaat nu niet 1 waarde teruggeven, maar een aantal waarden.

**Slide 59:** Ander vb: retrieve all supplier names who can supply at least one product with a green color. Meerdere queries binnen elkaar nesten. Query:

SELECT SUPNAME FROM SUPPLIER

WHERE SUPNR IN

(SELECT SUPNR

FROM SUPPLIES

WHERE PRODNR IN

(SELECT PRODNR

FROM PRODUCT

WHERE PRODCOLOR = ’GREEN’));

**Slide 60:** Nog een ander voorbeeld: select the product names that both supplier number 32 as supplier number 84 can supply. Query:

SELECT PRODNAME FROM PRODUCT

WHERE PRODNR IN

(SELECT PRODNR

FROM SUPPLIES

WHERE SUPNR = ’32’)

AND PRODNR IN

(SELECT PRODNR

FROM SUPPLIES

WHERE SUPNR = ’84’);

Dit zijn 2 blokken die eigenlijk perfect in parallel kunnen opgelost worden.

**Slide 62:** We kunnen hier veel geavanceerder in tewerk gaan aan de hand van gecorreleerde geneste queries. Hierbij gaat uw inner block refereren naar een tabel van het outer block waardoor die niet langer oplosbaar wordt.

**Slide 63:** Bv: retrieve the product numbers with at least two orders. Query:

SELECT P.PRODNR

FROM PRODUCT P

WHERE 1 <

(SELECT COUNT(\*)

FROM PO\\_LINE POL

WHERE P.PRODNR = POL.PRODNR);

Je begint ook hier met het inner block: het systeem loopt hierbij vast want er wordt gerefereerd naar tabel P die in het inner block niet gekend is. Enkel POL is daar gekend. Het systeem gaat dan automatisch kijken of tabel P gedefini¨eerd is in het outer block. Dat is het geval en nu gaat het systeem de eerste tuple selecteren en daarvoor gaat het het subblok oplossen, dan het tweede tuple enzovoort. Stel dat het eerste productnummer 0117 is, dan gaat het de inner block oplossen voor P.PRODNR = ’0117’. Het gaat alle purchase order lijnen checken op 0117. indien strikt groter dan 1, dan wordt 0117 gerapporteerd als antwoord.

**Slide 64:** Tabel bij query.

**Slide 65:** Ander voorbeeld: moeilijkste query van de cursus: retrieve number and name of all the suppliers who can supply a product with a price lower than the average price of that product; also include the number and name of the products, the purchase price and the delivery period. Query:

SELECT R.SUPNR, R.SUPNAME, P.PRODNR, P.PRODNAME, S1.PURCHASE\_PRICE, S1.DELIV\_PERIOD

FROM SUPPLIER R, SUPPLIES S1, PRODUCT P

WHERE R.SUPNR = S1.SUPNR

AND S1.PRODNR = P.PRODNR AND S1.PURCHASE\_PRICE <

(SELECT AVG(PURCHASE\_PRICE)

FROM SUPPLIES S2

WHERE P.PRODNR = S2.PRODNR) ORDER BY R.SUPNR;

Het systeem zal dus een zeer grote tabel maken -een JOIN-tabel- met allerlei koppelingen en het zal dan kijken of de purchase price voor een bepaalde tuple strikt kleiner is dan de purchase price van de supplies tabel (waarbij het om hetzelfde product moet gaan). Het is een gecorreleerde query want het refereert naar de tabel van de outer block.

**Slide 67:** Tabellen bij vorig voorbeeld.

**Slide 68:** Nog een voorbeeld: retrieve the three highest product numbers from the product table. Voor elk product gaan we het aantal producten met een hoger productnummer tellen. Als dit nummer kleiner is dan 3, behoort dit product tot de drie hoogste productnummers. Query:

SELECT P1.PRODNR

FROM PRODUCT P1

WHERE 3 >

(SELECT COUNT(\*)

FROM PRODUCT P2

WHERE P1.PRODNR < P2.PRODNR);

**Slide 16.2:** (Slide ontbreekt weer, zie Figuur 16.2) Tabel bij query. Als je bijvoorbeeld voor P1.PRODNR = 10 neemt (dus eerste uit tabel), dan gaat die inner block het aantal productnummers checken dat strikt groter is dan 10. Zo doe je dat voor elk productnummer en dan ga je kijken naar degenen met 0, 1, 2 grotere productnummers, dat zijn de resultaten die je wil.

**Slide 70:** ALL- en ANY-constructies:

* ALL zal waar opleveren indien, in dit voorbeeld (v *>* ALL V), de waarde van v strikt groter is dan alle waarden in de set V. Als de geneste query geen waarde teruggeeft, evalueert ALL tot ’waar’.
* ANY zal waar opleveren indien, in dit voorbeeld (v ¿ ANY V), minstens 1 waarde groter is dan ten minste 1 waarde in de (multi)set V. Als de geneste query geen waarde teruggeeft, evalueert ANY tot ’vals’. ANY is dus equivalent aan de IN operator.

**Slide 71:** Voorbeeld met ALL: retrieve the names of the suppliers who charge the highest price for a specific product. Query:

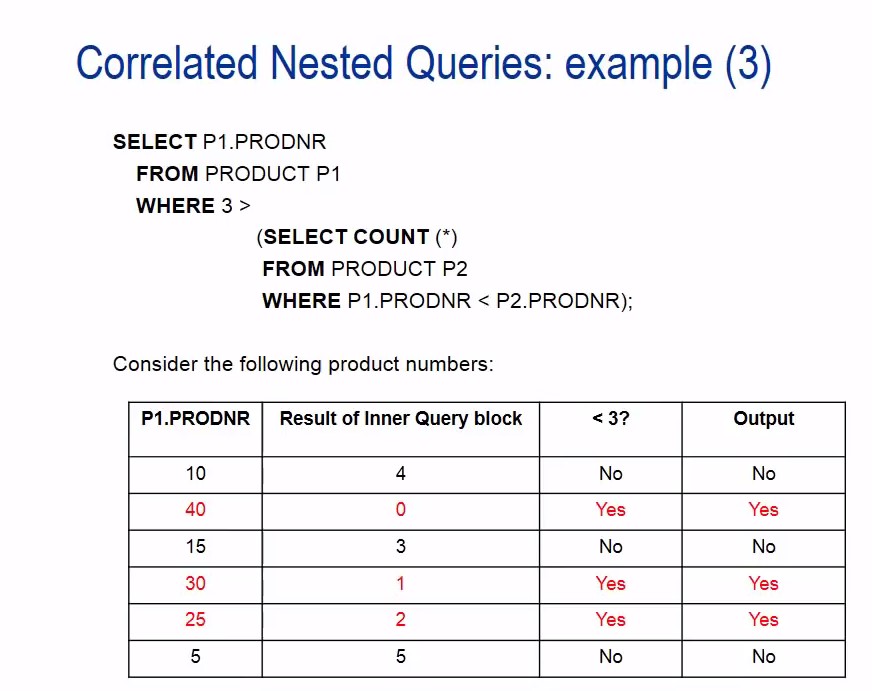


Figure 16.2: Ontbrekende slide.

SELECT SUPNAME FROM SUPPLIER

WHERE SUPNR IN

(SELECT SUPNR

FROM SUPPLIES

WHERE PRODNR = ’1245’

AND PURCHASE\_PRICE >= ALL

(SELECT PURCHASE\_PRICE

FROM SUPPLIES

WHERE PRODNR = ’1245’));

**Slide 73:** Tabellen bij query.

**Slide 74:** Voorbeeld met ALL (2): retrieve number and name for each supplier who has the highest evaluation code of all the suppliers located in the same city. Query:

SELECT R1.SUPNR, R1.SUPNAME, R1.SUPSTATUS

FROM SUPPLIER R1

WHERE R1.SUPSTATUS >= ALL

(SELECT R2.SUPSTATUS

FROM SUPPLIER R2

WHERE R1.SUPCITY = R2.SUPCITY);

**Slide 77:** Voorbeeld met ANY: retrieve the names of the suppliers who do not charge the lowest price for a certain product: je zoekt naar de naam van een supplier die niet de laagste prijs aanrekent en dan zoek je suppliers die nog meer aanrekenen (maar die eerste voldeed ook al!). Query:

SELECT SUPNAME FROM SUPPLIER

WHERE SUPNR IN

(SELECT SUPNR

FROM SUPPLIES

WHERE PRODNR = ’0117’ AND PURCHASE\_PRICE > ANY

(SELECT PURCHASE\_PRICE

FROM SUPPLIES

WHERE PRODNR = ’0117’));

**Slide 79:** EXISTS: gaat kijken of het resultaat van een subquery een waarde oplevert of niet. Het kijkt gewoon of die subquery een waarde oplevert, welke waarde interesseert EXISTS niet.

**Slide 80:** Vb: retrieve the names of the suppliers who can supply a specific product. Query:

SELECT SUPNAME

FROM SUPPLIER R

WHERE EXISTS

(SELECT \*

FROM SUPPLIES S

WHERE R.SUPNR = S.SUPNR

AND S.PRODNR = ’0117’);

Je gaat voor elke supplier kijken of het inner block iets oplevert en zo kijken of je de naam moet tonen.

**Slide 83:** For each supplier who can supply all products, retrieve name, address and city. Voor een supplier die alle producten kan leveren, bestaat er dus geen product dat hij niet kan leveren. Query:

SELECT SUPNAME, SUPADDRESS, SUPCITY

FROM SUPPLIER R

WHERE NOT EXISTS (SELECT \*

FROM PRODUCT P

WHERE NOT EXISTS (SELECT \*

FROM SUPPLIES S

WHERE R.SUPNR = S.SUPNR

AND P.PRODNR = S.PRODNR));

Gecorreleerde query want P en R zijn niet gekend (in binnenste blok). Per leverancier wordt er gekeken welk product niet geleverd kan worden en indien er geen enkel product niet geleverd kan worden, kan hij alles leveren en rapporteren we hem.

**Slide 85:** Overzicht van operatoren.

**Slide 86:** SQL heeft de unie, intersectie en except. Elk van deze kan je uitbreiden met ALL (zal duplicaten weghalen).

**Slide 87:** Voorbeeld: retrieve number and name of suppliers who are located in a certain city or can supply a specific product (or satisfy both conditions). Query:

SELECT SUPNR, SUPNAME

FROM SUPPLIER

WHERE SUPCITY = ’BRUSSELS’

UNION

SELECT SUPNR, SUPNAME

FROM SUPPLIER R, SUPPLIES S

WHERE R.SUPNR = S.SUPNR

AND S.PRODNR = ’0117’

ORDER BY SUPNAME ASC;

**Slide 88:** Voorbeeld met intersectie: retrieve number and name of suppliers who are located in a certain city and can supply a specific product. Query:

SELECT SUPNR, SUPNAME

FROM SUPPLIER

WHERE SUPCITY = ’BRUSSELS’

INTERSECT

SELECT SUPNR, SUPNAME

FROM SUPPLIER R, SUPPLIES S

WHERE R.SUPNR = S.SUPNR

AND S.PRODNR = ’0117’

ORDER BY SUPNAME ASC;

**Slide 89:** Met except: retrieve the number of the suppliers who cannot supply any product at the moment. Query:

SELECT SUPNR

FROM SUPPLIER

EXCEPT

SELECT SUPNR

FROM SUPPLIES;

EXCEPT == verschiloperator in sets.

**Chapter 17**

# Huistaak: Oefeningen SQL

## 17.1 Word document: Questions SQL StudentVersion

SQL The first exercises are based on the relations below. You can use the diagram as a reference. It is also possible to execute the queries (without installing anything). We highly recommend you to do so. In order to execute the queries, please go to: *http* : *//www.w*3*schools.com/sql/trysql.asp*?*filename* = *trysqlselectall* If you can’t see or do anything on the website, please click “restore database”.

**Question 1** Write an SQL query to show the name of all products from the “Condiments” category. Result: 12 records Answer:

SELECT P.ProductName

FROM Products P, Categories C

WHERE P.CategoryID = C.CategoryID

AND C.CategoryName = ’Condiments’;

Alternatieven: zie Toledo

**Question 2** Write an SQL query to show a list of countries where at least 10 customers are living. Display the country name and the number of customers.

Result: 3 records

Answer: Geen conditie op de ongegroepeerde data, wel op de gegroepeerde!

SELECT Country, count(\*) AS Nr

FROM Customers

GROUP BY Country

HAVING count(\*) >= 10;

WHERE wordt uitgevoerd voor GROUP BY dus kan geen conditie leggen op gegroepeerde data. HAVING wel, dus gaan we dat gebruiken!

**Question 3** Write an SQL query to show a list of customers that live in the countries where at least 10 customers are living. Display the name of the customer. Tip: you can reuse the previous query as a subquery. Result: 35 records Answer:

SELECT CustomerName

FROM Customers

WHERE Customers.Country IN (SELECT Country

FROM Customers

GROUP BY Country

HAVING count(\*) >= 10);

Je krijgt meerdere elementen terug (een verzameling), daarom gebruiken we geen ”=”.

**Question 4** Given the relation Orders and the relation Shippers. Assume that the relations are populated with the following values: Orders

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OrderID | CustomerID | EmployeeID | OrderDate | ShipperID |
| 1 | 90 | 5 | 1996-07-04 | 1 |
| 2 | 81 | 6 | 1996-07-05 | 1 |
| 3 | 34 | 4 | 1996-07-08 | 1 |
| 4 | 84 | 3 | 1996-07-08 | 1 |
| 5 | 76 | 4 | 1996-07-09 | 2 |
| 6 | 34 | 3 | 1996-07-10 | 2 |
| 7 | 14 | 5 | 1996-07-11 | 2 |
| 8 | 68 | 9 | 1996-07-12 | 2 |
| 9 | 88 | 3 | 1996-07-15 | 2 |
| 10 | 35 | 4 | 1996-07-16 | 2 |

Table 17.1: Orders

|  |  |  |
| --- | --- | --- |
| ShipperID | ShipperName | Phone |
| 1 | Speedy Express | 9831 |
| 2 | United Package | 3199 |
| 3 | Federal Shipping | 9931 |

Table 17.2: Shippers

How many results will the following queries return:

1. SELECT \*

FROM Orders, Shippers; Answer: 10 \* 3 = 30

1. SELECT \* FROM Orders

INNER JOIN Shippers ON (Orders.ShipperID = Shippers.ShipperID);

Answer: 10

1. SELECT \* FROM Orders

LEFT JOIN Shippers ON (Orders.ShipperID = Shippers.ShipperID); Answer: 10

1. SELECT \* FROM Orders

RIGHT JOIN Shippers ON (Orders.ShipperID = Shippers.ShipperID); Answer: 10 + 1 = 11

**Question 5** Given query 1, 2 and 3 and two relations Person(Id, Name, FunctionId) and Function(FunctionId, FunctionName) with each at least two records.

1. SELECT \*

FROM Person, Function;

1. SELECT \*

FROM Person, Function

WHERE Person.FunctionId = Function.FunctionId;

1. SELECT \* FROM Person

INNER JOIN Function ON (Person.FunctionId = Function.FunctionId);

Which of the following is true:

1. Query 2 will always return more results than query 1.
2. Query 2 will return exactly as many results as query 3
3. Query 2 will return more results as query 3
4. Query 2 will return less results as query 3 Answer: 2. is correct.

**Question 6** Assume you want to return a list of countries and the number of suppliers living in that country. Which of the following is true:

1. You have to use SELECT DISTINCT
2. You have to use GROUP BY → want we willen groeperen per land
3. You can use SELECT DISTINCT or GROUP BY: they can be used to show exactly the same result.
4. It is impossible to return such a list using SELECT DISTINCT or GROUP BY. You will have to use a subquery. → het kan ook zonder subquery!

Answer: 2. is correct.

**Question 7** Given the following SQL query:

SELECT supname, supaddress, supcity

FROM supplier r

WHERE NOT EXISTS (SELECT \*

FROM product p WHERE EXISTS

(SELECT \*

FROM supplies s

WHERE r.supnr = s.supnr

AND p.prodnr = s.prodnr

)

);

This query selects:

1. The supplier name, supplier address and supplier city of all suppliers who cannot supply any products.
2. The supplier name, supplier address and supplier city of all suppliers who cannot supply all products.
3. The supplier name, supplier address and supplier city of all suppliers who can at least supply 1 product.
4. The supplier name, supplier address and supplier city of all suppliers who can supply all products.

Answer: 1. is correct.

**Question 8** Given the following query:

SELECT supname, supaddress, supcity

FROM supplier r

WHERE EXISTS

(SELECT \*

FROM product p WHERE EXISTS

(SELECT \*

FROM supplies s

WHERE r.supnr = s.supnr

AND p.prodnr = s.prodnr

)

);

This query selects

1. The supplier’s name, address and city of the suppliers who can’t supply any product.
2. The supplier’s name, address and city of the suppliers who can supply all products.
3. The supplier’s name, address and city of the suppliers who can supply at least one product.
4. The supplier’s name, address and city of the suppliers who can’t supply all products.

Answer: 3. is correct.

**Question 9** The following two relations are from the purchase administration database (primary keys are underlined, foreign keys are in italics):

SUPPLIER(SUPNR, SUPNAME, SUPADDRESS, SUPCITY, SUPSTATUS) PURCHASE ORDER(PONR, PODATE, *SUPNR*)

Given the following question:

Select all supplier numbers, together with their supplier name and total number of outstanding orders for each supplier. Include all suppliers in the result, even if there are no outstanding orders for that supplier at the moment. Which SQL query is CORRECT?

1. SELECT s.supnr, s.supname, count(\*) AS outstanding\\_orders

FROM supplier s

RIGHT OUTER JOIN purchase\\_order AS po

ON (s.supnr = po.supnr) GROUP BY s.supnr;

1. SELECT s.supnr, s.supname, count(po.ponr) AS outstanding\\_orders

FROM SUPPLIER s

FULL OUTER JOIN purchase\\_order AS po ON (s.supnr = po.supnr);

1. SELECT s.supnr, s.supname, count(\*) AS outstanding\\_orders

FROM supplier s

LEFT OUTER JOIN purchase\\_order AS po

ON (s.supnr = po.supnr) ORDER BY s.supnr;

1. SELECT s.supnr, s.supname, count(po.ponr) AS outstanding\\_orders

FROM supplier s

LEFT OUTER JOIN purchase\\_order AS po

ON (s.supnr = po.supnr) GROUP BY s.supnr;

Answer: 4. is correct.

INNER LEFT/RIGHT JOIN: alleen de gematchte rijen geven, geen NULLwaarden.

**Chapter 18**

# Les 15: 30/03/2015

## 18.1 Slides: 1cITGovernance

**Slide 1:** IS Strategy: hoe IS’en bedrijven kunnen be¨ınvloeden. IT governance is heel belangrijk.

**Slide 4:** Corporate governance. Enron-schandaal/WorldCom: WorldCom was een heel groot bedrijf uit de VS, de Telenet van de VS. Enron was een energiebedrijf. Beide bedrijven kwamen rond 2000-2002 in het nieuws. Ze zijn illustratief voor accounting fraud: fraudulent activities. Ze weerspiegelen een financi¨eel wanbeheer dat samen met de Internetbubble explodeerde. Een aantal bedrijven wou dus buiten de wet om financi¨ele fraude plegen om hun marktwaarde op te krikken. Enron: miljarden dollars aan valse winst die gerapporteerd werd. Aan de hand van buiten-balansactiviteiten: offshore bedrijven buiten de VS maken, die schulden laten aanmaken voor Enron zodat zij winst leken te maken. Niemand wist van die bedrijfjes. De CFO trachtte die schulden buiten het bedrijf te houden. Op die manier werd het marktaandeel van Enron steeds hoger geduwd. De board of directors werden bijna omgekocht aan de hand van die stijgende aandelenkoers, dat was het grootste probleem. Vanuit deze schandalen zijn er nieuwe wetgevende initiatieven gekomen: strictere regulering wat betreft accounting en controle (niet alleen financi¨eel, maar ook op plichten wijzen om controleactiviteiten op te zetten zodat de CFO, CEO en the board of directors effectief kunnen aantonen dat er zeker geen fraude wordt gepleegd).

**Slide 5:** Er werden heel sterke, nieuwe wetgevende initiatieven ingevoerd. Een van de belangrijkste is de Sarbanes-Oxley Act: vbelangrijkste wet die er na de schandalen gekomen. Door 2 senatoren: Sarbanes en Oxley: nieuwe wetgeving om bedrijven strikter op te leggen hoe ze hun controleactiviteiten moeten organiseren: corporate governance: je zaken beheren, welke regels je moet volgen. SOX werd gestemd door de Amerikaanse senaat. De belangrijkste elementen: leiders van een bedrijf (board of directors, CEO, CFO) zijn persoonlijk verantwoordelijk voor fraude. Als er dus foutieve info wordt gecommuniceerd naar de buitenwereld, kunnen die mensen persoonlijk worden gedagvaard en veroordeeld. Voor SOX kwamen heel veel van die leidinggevenden af met dat ze van niks weten, dat het lager kader daarvoor instond. → Eerste puntje.

Ook: de rol/functie van externe auditoren. Arthur Anderson: zat mee in die fraude als externe auditor: hebben de frauduleuze statements geconfirmeerd. De rol werd duidelijker gedefini¨eerd van hoe externe auditors hun job moeten uitvoeren en welke mate van onafhankelijkheid zij moeten uitvoeren. Voordien was het mogelijk dat de leidinggevende van een bedrijf later naar een auditingbedrijf ging dat dan nauw samenwerkte met zijn vorig bedrijf → maakt dat fraude makkelijker mogelijk is.

Derde: Section 404: gaat over het concept van een interne controlestructuur: bedrijven in de VS die publiek verhandeld worden, worden verplicht om te kunnen aantonen dat zij een structuur op poten gezet hebben die ervoor zorgen dat zij die controleactiviteiten kunnen uitvoeren. Bedrijven worden aan de hand van die wet verplicht om een controlestructuur op poten te zetten zodat ze kunnen aantonen dat financi¨ele info correct is, ze geen fraude kunnen plegen en dat de interne structuur correct is. Zorgt ervoor dat controles uitgevoerd kunnen worden. Is een heel organisatiebrede policy die moet ingevoerd worden om die controleactiviteiten op te kunnen zetten. Daarover gaat corporate governance. Is de reden dat het zo belangrijk is dat IT eraan gekoppeld wordt.

SOX heeft ook Europese wetgevende initatieven ge¨ınspireerd. In Belgi¨e: Code Lippens: corporate governance raamwerk (niet echt een wetgeving, in de VS wel), best practices die zorgen voor gelijkaardige reguleringen als in de VS. Ook in Europa moet je kunnen aantonen dat je de interne controlestructuur hebt opgezet.

**Slide 6:** Zegt waar governance over gaat. Controle kunnen nagaan van compliance met wetgeving. Het algemene plaatje ziet eruit zoals op deze slide. Een bedrijf wordt geleid door een management, maar alles start bij de stakeholder needs en voornamelijk de shareholder needs; niet altijd, maar aandeelhouders van een bedrijf beslissen gezamelijk hoe een bedrijf eruit moet zien. Is in samenspraak met de board of directors uiteraard. Zij zetten de richtlijnen over hoe een bedrijf moet geleid worden, wat de doelstellingen van een bedrijf moeten zijn. Vaak: winst genereren. Wat de stakeholders willen, is de drive. Het objectief is meestal om winst te cre¨eren en governance houdt in om dat te structureren. Corporate governance gaat erom om die stakeholder needs om te zetten in benefits die gerealiseerd worden (en wie ervan gaat kunnen genieten), niet teveel risico’s nemen en resource optimisation: wie/wat is noodzakelijk om value te bereiken? Wat is de juiste inverstering in resources? Die 3 vormen samen wat je moet uitvoeren om de strategie van de shareholders uit te voeren. Leidt tot het punt dat je als organisatie in staat moet zijn om te controleren, organiseren. Die controleinfrastructuur noemen we internal control: kunnen nagaan of je bepaalde doelstellingen haalt, of je compliant bent met wetgeving.

**Slide 8:** Internal control: je moet in staat zijn om te kunnen garanderen (’reasonable assurance’) tegenover de te behalen objectieven. Die objectieven komen van de stakeholders. Doelen:

* Effectiveness and efficiency of operations: het gaat niet enkel over het tegemoet komen aan de wetgeving van het land waarin je actief bent, het is veel breder dan dat. Kan je de effectiviteit en effici¨entie van jouw interne processen garanderen? Je moet in staat zijn om dat na te gaan! Heb je de juiste info in handen als CEO om te kunnen weten dat jouw bedrijf op een goede manier werkt?
* Reliability of financial reporting: kunnen garanderen dat alle financi¨ele info die je beschikbaar moet stellen, dat die correct/betrouwbaar is. Ook wat dat betreft garanties kunnen geven.
* Compliance with applicable laws and regulations: voldoen aan de wetgevingen waar je aan moet voldoen. Maar corporate governance is breder dan dat. Je moet ook compliance kunnen aantonen met wetgevingen, maar de 2 bovenstaande puntjes zijn dus ook belangrijk.

→ Interne controle.

Om als bedrijf te kunnen garanderen dat je die interne controlestructuur hebt en dat die correct werkt en goed functioneert, maak je gebruik van standaarden die door veel bedrijven worden toegepast.

**Slide 9:** Je gaat op zoek naar raamwerken, standaarden die beschrijven hoe je zo’n interne controlestructuur moet opzetten en hoe je kan beschrijven in documenten hoe je dat implementeert. Het belangrijkste raamwerk hiervoor is waarschijnlijk COSO Internal Control - Integrated Framework: belangrijkste corporate governance raamwerk dat er bestaat.

COSO: zie eerste bullet. Zitten heel veel grote bedrijven achter: PWC (een van de initiatiefnemers, hebben mee aan de kar getrokken om dat framwerk op te bouwen en het raamwerk door te drukken en te zorgen dat andere bedrijven er mee in stappen). Dit is het belangrijkste raamwerk om die SOX compliance te garanderen (sus zien dat je die SOX-wetgeving volgt).

COSO-cube: wat je doet om te kunnen aantonen dat je SOX-compliant bent. Je neemt die standaard erbij, daarin staat beschreven hoe je als bedrijf moet aantonen en organiseren dat je SOX-compliant bent. Gaat heel diep: hoe controlestructuren in elkaar zetten, activiteiten die je opzet,... → Aan de hand van maturiteitsscores aangeven hoe goed je bent in bepaalde zaken. Je gaat in detail beschrijven hoe alles in elkaar zit. Je krijgt dan een heel lijvig document dat voor uw bedrijf vertelt hoe dat ge¨ımplementeerd is. (paper over COSO vs COBIT gewoon lezen, niet leren!)

De COSO-cube is een algemene beschrijving: er zijn objectieven, componenten van je structuur en de structuur van de organisatie. Onderdelen van de cube:

* Objectieven: operations, reporting and compliance: COSO-definitie van internal control. Die worden telkens uitgesplitst. Compliant: wetgeving volgen. Die worden telkens verfijnd. Begint heel high-level maar wordt telkens verfijnd en uitgediept in subdoelstellingen.
* Structuur: je gaat die oefening maken voor elk niveau van de organisatie. Je begint op het hoogste niveau: entity level en dan duw je dat door naar de divisies, operational units, bepaalde functies binnen de organisatie. Het blijft dus niet op het managementniveau gesitueerd.
* Component (belangrijkste!): wat zijn de componenten van het integrated framework: control environment, risk assessment,... → Staan heel mooi beschreven in dat document.
  + Control environment: gaat voornamelijk over het scheppen van de juiste sfeer binnen een organisatie wat betreft interne controle: iedereen binnen de organisatie moet zich bewust zijn van het feit dat hij/zij gecontroleerd wordt. Je zorgt ervoor dat niemand binnen de organisatie zich er niet van bewust is dat hij gecontroleerd wordt, iedereen moet plichtsbewust zijn.
  + Daarbij aansluitend: control activities: opzetten van de juiste activiteiten om te controleren. Heel sterk verbonden met control environment. Typische voorbeelden: segregation of duties.
  + Risk assessment: welke risico’s nemen we als organisatie en hoe managen we die risico’s? Risico-management; op vlak van IT, investeringen,... → Risico’s correct beschrijven en inschatten en bewust een keuze maken of je die risico’s aangaat en hoe je ermee omgaat.
  + Information & communication: beschikt jouw bedrijf en elke functie binnen jouw bedrijf over de juiste info om beslissingen te nemen en communiceren de mensen binnen de organisatie op de correcte manier?
  + Monitoring activities: tools en dashboards die toelaten om compliance en (nog iets) op te volgen. Effectief controleren van wat er aan de gang is binnen het bedrijf.

Wat je ermee moet doen is onduidelijk op zich, maar je vult dat in voor jouw bedrijf.

**Slide 10:** SOX Compliance: COSO framework wordt daarvoor gebruikt. Corporate governancesystemen zonder IS is zeer moeilijk/onmogelijk. IS’en zijn cruciaal om die internal control structure, corporate governance te kunnen doen. Wat we zien is dat bv COSO enkel in het 4e puntje aangeeft dat informatie belangrijk is in het components deel, maar is dus wel cruciaal. Mensen zijn dus op de proppen gekomen met meer IT-gerichte raamwerken. Zodra bedrijven nadenken over controle, effici¨entie, kom je heel snel in aanraking met IT: hoe gaan we die data capteren, rapporteren,... → IT-governance frameworks: meer IT-gefocused: hoe je IS’en moet opzetten om corporate governance te garanderen. Voor IT governance: COBIT. Doet in principe wel hetzelfde als eender welk raamwerk. Hier nemen we IS als startpunt om te beschrijven hoe een bedrijf voldoet aan de SOX-act.

**Slide 12:** Zelfde idee als corporate governance, alleen vanuit een IT-perspectief: je moet kunnen aantonen dat je de interne controlestructuur hebt. En het is de verantwoordelijkheid van de board of directors, maar ook van iedere andere functie binnen het bedrijf.

**Slide 13:** Heel sterke link met IT strategie. Een IS is een carbon copy van jouw realiteit als organisatie. Elk proces dat je in de realiteit uitvoert (verkopen,investeren,...) wordt weerspiegeld in het IS. En net omdat de realiteit gekopi¨eerd wordt, is jouw IS een perfect bron, een perfect startpunt om die corporate governance aan te tonen.

**Slide 14:** COBIT: ISACA heeft die opgesteld (niet te kennen voor examen), professionele associatie van mensen die IT-governance doen.

Practice-driven: raamwerken zijn heel interessant, worden heel frequent gebruikt en veel organisaties moeten daarop vertrouwen. Er is een heel grote business rond, maar die framworks komen uit de praktijk: er zijn mensen die samengezeten hebben en dat geschreven hebben. Maar dat is nooit wetenschappelijk gevalideerd: die standaarden zijn ontstaan binnen de industrie, maar die raamwerken zijn nooit wetenschappelijk onderbouwd. Wat erin staat, moet je altijd met een korrel zout nemen. De zaken die beschreven zijn, zijn dus niet wetenschappelijk onderbouwd, dus geen garantie dat organisatie perfect gaat lopen. De zaken die daarin staan komen voort vanuit de cases die er zijn in de praktijk.

**Slide 15:** COBIT 5: heel ruime standaard. Bestaat uit 10-11 chapters en volgt de 5 principles van COBIT (te zien op slide):

1. Meeting stakeholder needs: beschrijft dat je de stakeholder needs moet tegemoet komen en hoe je dat doet. Bv Cole’s cascade: jouw stakeholders zetten objectieven en die worden gespecifi¨eerd naarmate je dieper gaat in de organisatie. Je moet als organisatie in staat zijn om dat ook te doen.
2. Alle processen binnen jouw bedrijf aanpakken (zie ook **Slide 17**): COBIT is heel hollistisch en breed, dekt alles binnen de organisatie af: governance van een bedrijf en alle high-level processen die nodig zijn. Maturiteitsmodel **Slide 18**: scores om te zeggen wat betreft bv managen van innovation, hoe goed je organisatie het doet.
3. Organisatie volledig afdekken en COBIT laat toe dat je andere standaarden kan inpluggen in COBIT. Er zijn veel specifiekere standaarden (bv ISO-normen: kwaliteitsstandaarden) en deze zijn dus makkelijk in te pluggen in COBIT.
4. Sluit aan bij covering end-to-end en 3.
5. Strikte opdeling tussen de verantwoordelijkheid van board of directors en de verantwoordelijkheid van managers. De uiteindelijke verantwoordelijkheid ligt dus bij de board of directors. De governance functie ligt bij de board of directors, niet bij het management. Alles wat met governance te maken heeft ligt dus bij board of directors (zie **Slide 16**).

**Slide 19:** Raamwerken.

## 18.2 Slides: 5. CustomerRelationShipManagement

**Slide 3:** Eerste 3 applications zijn het belangrijkste.

**Slide 4:** CRM: het is een strategie, realisatie ervan met het doel om customer return en satisfaction te maximaliseren. Je imago is tegenwoordig bijna even belangrijk als het genereren van extra omzet. Soms is het zelfs ondenkbaar/onnuttig om extra te verkopen als dat zorgt voor minder klanttevredenheid. De laatste jaren is het plaatje volledig omgedraaid: in plaats van producten meer te marketten om meer te verkopen zijn we gegaan naar het verbeteren van de perceptie van de producten. Klanttevredenheid is van primordiaal belang geworden. De return is nog steeds belangrijk, maar tevredenheid van klanten wordt steeds belangrijker, ook omwille van sociale media onder andere.

**Slide 5:**

* Klanten eisen steeds meer en zijn mondiger (wegens social media).
* Klanten kunnen producten kopen in zowat elke hoek van de wereld, dat wilt zeggen dat er veel meer producten beschikbaar zijn, dus het is ook veel makkelijker om een andere relatie aan te gaan, om andere producten te kopen.
* Mass customization: klanten eisen steeds meer een persoonlijke aanpak, iets dat getailored is naar jouw gebruiksprofiel,... Specifiek gericht om jouw klantenervaring uniek te maken. Is soms artifici¨eel, maar het gaat erom dat je jouw dienstverlening, producten zo laat zijn dat klanten denken dat het product uniek voor hen bestaat. Met andere technologie wordt het ook mogelijk om meer te customiseren.
* Je moet je klanten zo goed kennen dat je perfect weet hoe je ze moet bedienen etc.
* Laatste 2 puntjes samen: klanten worden steeds mondiger, zijn steeds meer mogelijk om te communiceren over ervaringen en ze doen dat ook. Enkele slechte tweets over jouw producten kunnen een enorme impact hebben op jouw bedrijf. Het voorlaatste puntje bestond al, maar het grootste probleem nu is dat customers kunnen gaan klagen voor heel de wereld, dus heeft het meer impact.
* Zelfs als klanten ontevreden zijn, krijg je als bedrijf nog altijd een kans om het recht te zetten. Je kan er vaak in slagen om op een positieve manier met klachten om te gaan, door daar open over te communiceren, kan je als bedrijf een extra element realiseren. Je beseft dat je klanten ontevreden zijn, weet waarom en je tracht ze heel snel te helpen. → Imago van het bedrijf verbetert, hoewel de service initi¨eel als slecht werd bevonden. Als er dus slechte commentaar gegeven wordt, moet je dat heel snel spotten en ervoor zorgen dat je mensen hebt om die personen te contacteren en te zorgen dat die klanten geholpen worden en daar open over communiceren.

**Slide 6:** Waar marketeers vroeger meer ge¨ınteresseerd waren in meer verkopen, is relatie met de klant nu belangrijker.

**Slide 7:** Heel veel bedrijven maken gebruik van enorm veel data over hun klanten, om zo hun klanten te leren kennen of om dienstverlening beter te maken. Daarvoor dient dan ook analytics: meer inzicht krijgen in waarom een klant bv het bedrijf verlaat.

**Slide 11:** Retention modeling/churn prediction: cre¨eren van modellen die toelaten om klanten te behouden. Churn: klantenverloop. Als je aan retention modeling doet, kan je dat ook gebruiken om aan churn prediction te doen. Je gaat modellen bouwen, analytische modellen, die verklaringen geven voor het feit dat klanten jouw bedrijf verlaten. Retention is belangrijk omdat:

* Langetermijnklanten gaan zich minder en minder beuwst zijn van de prijzen die zij betalen voor jouw producten, ze gaan makkelijker bereid zijn om hogere prijzen te betalen.
* Langetermijnklanten zijn minder kostelijk om te bedienen.
* Aggregatie van de eerste twee: klanten die meer bereid zijn om te betalen en minder kosten, brengen dus meer op: hogere lifetime value.

In de telco sector en relationship buyers, subscriptiecontracten. Transaction buyers: mensen die 1 keer kopen, heb je veel minder een relatie (bv auto kopen) mee.

**Slide 12:** Mobistar en Netflix: kwamen allebei recent onder druk omwille van churn. Mobistar: tot in 2014 5-6 jaar enorm afgezien omdat zij een gigantisch hoog klantenverloop hadden. Hun aandeel op de Brusselse beurs is erg omlaag gegaan. Sinds 2014 realiseren ze weer betere churn rates. Netflix: ook een subscriptiegebaseerd bedrijf. Introduceerden zich in Belgi¨e en Nederland. Werd heel snel duidelijk dat Netflix niet makkelijk haar doelstellingen kon bereiken wat betreft klantenacquisitie, zie ook volgende slide.

**Slide 13:** De aandelen van Netflix zijn gezakt omdat de expansie naar Europa minder succesvol was dan gedacht. Dit komt omdat customer acquisition veel moeilijker is dan customer retention. Als je een bestaand bedrijf bent in een bepaalde markt, moet je besefffen dat het veel makklijker/beter is om je geld te steken in bestaande klanten dan in het aantrekken van nieuwe klanten (tot 6 `a 7 keer duurder om nieuwe klanten aan te trekken).

**Chapter 19**

# Les 16: 03/04/2015

## 19.1 Slides: 5. CustomerRelationshipManagement

**Slide 14:** We willen modellen bouwen die bruikbaar zijn om te gaan voorspellen welke klanten het bedrijf zullen verlaten. Als we zo’n model hebben, kennen we de factoren die meespelen. We moeten aandacht hebben voor verschillende soorten van churn. 2 belangrijke categorisaties:

* Causaal effect:
  + Vrijwillige churn (deliberate): klanten waar we het meest in ge¨ıntersseerd zijn: klanten die uit zichzelf beslissen om jouw diesnt niet meer te gebruiken omwille van een of andere reden (die je liefst van al kent).
  + Forced: je kan als bedrijf beslissen om klanten op te zeggen. Voor sommige klanten zal je als bedrijf zelf kiezen om de relatie op te zeggen.
  + Expected: klantenverloop waarvan je het op voorhand kan verwachten. Bv: pampers.

We moeten in al die zaken het causaal effect in rekening gaan brengen. Je moet die twee laatste categorie¨en van klanten niet in rekening brengen; je bent voornamelijk ge¨ınteresseerd in klanten die vrijwillig jouw bedrijf verlaten. Ofwel weet je het al, ofwel wil je het zelf, dus weet je waarom klantenrelatie wordt stopgezet.

* Visibiliteit: het feit dat je niet altijd als bedrijf weet of een klantenrelatie al dan niet gestopt is of niet. In sommige gevallen is het heel duidelijk:
  + Active churn: event, bv brief, of feit in de realiteit waarvan je kan zeggen dat de klantenrelatie *nu* stopt, dan heb je dus active churn.
  + Passive churn: bv het feit dat je minder vaak gebruik gaat maken van je kredietkaart. Voor veel bedrijven is het onduidelijk of de klante beslist heeft om de dienst al dan niet nog te gebruiken, of dat het tijdelijk gestopt/minder is. Het is veel moeilijker te merken dan actieve churn.

**Slide 15:** Hoe churn van strategisch belang kan zijn: op een bepaald ogenblik beslisten de mensen van UPS om voor een ganse week in staking te gaan. FedEx zag toen een gigantische increase van pakketten/brieven die via hen verstuurd werden. Nadat UPS weer begon te werken ging dat terug naar beneden, maar opeens kende FedEx de klanten van UPS. FedEx is dus die klanten gaan identificeren en gaan targetten om te kijken waarom die klanten terugkeerden naar UPS. Ze hebben dus een specifieke campagne opgezet om nieuwe offers te doen naar die klanten. Vanuit bedrijfsperspectief kan churn dus zeer belangrijk zijn.

**Slide 16:** Hoe gaan we voorspellen welke klanten onze services niet meer gaan gebruiken? → Data mining en data analytics. We vertrekken van een heel aantal variabelen, waarvan ´e´en de predictieve variabele is (churn).

* Demografische data: leeftijd, burgerlijke staat, geslacht,...
* Product/service usage data: al heel sterk gerelateerd aan RFM data.
* Complaints data.
* (Social) network info: zie later.

**Slide 17:** Recency, Frequency and Monetary framework: theorie die al heel lang bestaat. Het gaat erom dat je info die je hebt in jouw operationele systemen die jouw klant beschrijven en die het gebruik van jouw klant beschrijven, dat je die gaat meten en defini¨eren. 3 categorie¨en dus: R, F, M:

**Recency:** Het aantal maanden sinds laatste aankoop: hoe recent was de laatste aankoop?,...

**Frequency:** Hoeveel keer bepaalde producten zijn gekocht binnen bepaalde tijdsperiodes. Heel veel operationalisaties van die frequency-dimensie kan je opnemen.

**Monetary:** Hoeveel spendeert jouw klant aan jouw product? Zeker als er een variabele kost is voor een klant is dit nuttig.

Deze variabelen worden heel vaak gebruikt om churn-prediction op te stellen. Veel variabelen kunnen enkel en alleen gemeten worden voor bestaande klanten. Het is dus heel moeilijk om die RFM toe te passen bij het zoeken naar nieuwe klanten.

**Slide 18:** Abstract voorbeeld om te tonen hoe zo’n dataset eruit ziet. Vanonder: churn-prediction model.

2 soorten variabelen: churn: variabele die we willen voorspellen. RFM: vaak veel verschillende soorten variabelen daarbinnen. 1 predictieve variabele en de 4 linkse variabelen zullen gebruikt worden in het model om de laatste variabele te gaan voorspellen. Customer zal uiteraard niet gebruikt worden voor churn want is uniek per klant, is gewoon ID.

**Slide 19:** Response modelling: nieuwe klanten gaan aantrekken. Kunnen we modellen bouwen zodat we nieuwe klanten kunnen aanwerven? → Customer acquisition.

**Slide 20:** Je wilt gaan optimaliseren aan wie je bv catalogi gaat sturen, gegeven dat je budget beperkt is. We gaan ervan uit dat het niet mogelijk is om met een onbeperkt budget te werken. We gaan dus trachten een model te bouwen dat toelaat die catalogi zo optimaal mogelijk te sturen, naar die klanten die het meest waarschijnlijk zijn om te gaan reageren, kopen. Ook hier gaan we dus een analytisch model bouwen op basis van data. Van RFM weten we hier heel weinig, maar in heel veel settings gaat het aantal mogelijke variabelen hier lager zijn dan bij churn.

**Slide 21:** Wat is de kans dat een klant een aankoop gaat doen na het sturen van een catalogus?

**Slide 22:** Score opstellen die aangeeft hoe waarschijnlijk het is dat de klant zal reageren op de marketingcampagne. We gaan een bepaalde selectie van klanten targetten (met de hoogste score) met de campagnes. Gegeven dat dit een heel grote dataset wordt en gegeven jouw budget, moeten we een optimale selectie maken van klanten die we gaan targetten (2000 of 100 000 klanten aanschrijven?).

**Slide 23:** Inzicht krijgen in de effecten die zo’n response model/net lift model cre¨eren: gain charts. Gain chart = een van de belangrijkste modellen om response modelling campagne op een kwalitatieve manier te analyseren. We gaan vergelijken ten opzichte van een random model/keuze.

**Slide 24:** Gains chart: het is een kwantitatieve analyse van een response model. Dat wordt afgebeeld door de donkerblauwe lijn. Gains chart gaat (de mogelijkheid bieden om) een vergelijking (te) maken tussen response models onderling maar ook ten opzichte van de case waarin je geen model gaat gebruiken (dus op random manier klanten selecteren). In het perfecte model ga je op voorhand een inschatting maken hoeveel klanten positief zullen reageren. → Moeilijk in te schatten. In dit geval is het 10%.

De horizontale as geeft het percentage van klanten aan dat je gaat targetten. Heel belangrijk om dit te gaan bekijken met absolute cijfers. Stel dat je in totaal 10 000 mogelijke klanten hebt, daarvan weten we dat er 1000 zullen reageren (zie perfecte model). De horizontale as geeft aan hoeveel procent van je klanten je een catalogus gaat versturen. 30% zegt bv dat je aan 3000 klanten een catalogus gaat sturen.

Verticale as: het percentage van het aantal potenti¨ele responders dat je effectief hebt getarget. Als je dus 30% van de populatie aanspreekt, gaan we 30% van de mensen aangesproken hebben die zouden gereageerd hebben (bij no model). Via response model: 10% mensen aanschrijven en 30% van het aantal mogelijke mensen dat zou reageren zal getarget zijn.[[1]](#footnote-1) Response model is de verbetering die gerealiseerd is door het model te gebruiken. Via response model zou je niet 300 klanten hebben getarget, maar 650.

Horizontaal: mensen naar wie je catalogus gaat sturen. Verticaal: geeft resultaat weer hoeveel klanten effectief gereageerd zouden hebben moest je ze allemaal aangeschreven hebben/moest je het perfecte model hebben. Je moet weten hoe groot jouw populatie is en een inschatting hebben van hoeveel respondenten er zullen zijn. Als je van 10 000 mensen hebt en je schrijft 50% aan, zal 85% reageren. → EXAMENVRAAG!

**Slide 25:** Je moet nog een analyse gaan doen van wat het oplevert, wat is de potenti¨ele opbrengst die je realiseert? Je moet rekening houden met de kost. Welk percentage gaan we selecteren om de kosten af te wegen van de marketingcampagne ten opzichte van de opbrengsten door die mensen te targetten? → Optimisatieprobleem.

**Slide 26:** Soorten responders:

* Persuadables: mensen die je kan overtuigen. Klanten die enkel zullen aankopen als je het marketinginitiatief neemt.
* Sure things: marketing is nutteloos voor hen: zij zullen sowieso kopen.
* Lost causes: die klanten die nooit zullen reageren/aankopen, of je nu marketing doet of niet.
* Do not disturbs or sleeping dogs: gevaarlijkste groep van klanten: die klanten die gaan reageren in de negatieve zin op jouw marketingcampagne: zij gaan beslissen om te churnen na een marketinginitiatief. Bv elektriciteitscontracten: na 15 jaar plots korting krijgen zonder reden: je gaat kijken naar alternatieven: je wordt jezelf bewust dat het goedkoper kan.

**Slide 28:** Verzamelnaam voor verschillende manieren om extra te gaan verkopen. X-selling: binnen jouw bestaande klantenbasis trachten extra te gaan verkopen/betere return te cre¨eren. Belangrijk zijn de definities hierbij: verschillende manieren van X-selling. Willen alledrie een betere return cre¨eren binnen bestaande klantenbasis:

* Up selling: een duurder/beter product verkopen: een dat jou meer gaat opbrengen. Bv frituur: een groter pak friet voorstellen.
* Cross selling: nog iets extra voorstellen: rechtstreeks/zeer sterk gerelateerd aan het product: iets verkopen dat niet exact het product zelf is, maar eraan gerelateerd is.
* Down selling: je gaat als bedrijf zeggen dat je iets best niet koopt: ”zou je die groene thee daar wel bij drinken?” Ook al koop je niks anders. Is als winkeluitbater beter: als klant groene thee zou drinken met frieten zou die waarschijnlijk niet terugkeren. Anders komt klant waarschijnlijk wel terug: slechte ervaring voorkomen.

Data mining: zoeken naar patronen:

* Welke producten worden vaak samen gekocht? Op basis van die patronen kan je initiatieven nemen om meer te verkopen.

**Slide 29:** X-selling wordt ook gebruikt online: bv Amazon: verkopen van extra items gegeven de interesse van een klant om een bepaald product te kopen: recommender systems: systemen die in staat zijn om te voorspellen wat de klant ook nog wilt, gegeven de huidige keuze.

Netflix organiseerde een wedstrijd voor data scientists om te gaan voorspellen of een bepaalde klant een bepaalde film leuk zou vinden. Het beste team/beste model kregen/kreeg $1 miljoen. Op basis van karakteristieken van een film werd dat bekeken. Hebben het zo kunnen configureren dat het bijna uniek is voor alle klanten.

**Slide 30:** Voorbeeld van Amazon.

**Slide 32:** Profilering van klanten om allerlei redenen. Zeer interessant om bv aan customisatie te doen. Segmentatie: opdelen van klanten in segmenten die klanten groeperen op zo’n manier dat klanten binnen 1 groep heel sterk op mekaar gelijken. Als de segmenten gedefini¨eerd zijn, moeten die zo ver mogelijk uit elkaar zitten, zodat ze zo veel mogelijk van elkaar verschillen. Binnen 1 segment wil je klanten heel sterk op mekaar laten gelijken en je wilt de verschillen tussen segmenten zo groot mogelijk. → Clustering. Is noodzakelijk: als je klanten hebt die heel hard op mekaar lijken, kan je initiatieven nemen die heel sterk aansluiten bij de wensen van die klanten. Als je er dan nog eens in staat bent om die segmenten uit elkaar te houden, weet je hoe je op de verschillen tussen die klanten kan gaan inspelen om jouw product beter af te stemmen op de wensen van de klanten. Voorbeelden: targeted marketing: gaat toelaten om folders op te stellen die een heel specifiek segment van jouw klantenbasis targeten. Beheren portfolio’s: optimaliseren in termen van welke merken je meot hebben, welke portfolio’s van producten je nodig hebt, wat er nog ontbreekt voor bepaalde segmenten. Je kan een veel betere dienstverlening/productportfolio gaan uitbouwen en een veel beter inzicht hebben in wat ontbreekt/moet verbeteren.

Winstgevendheid: als je wil inschatten welke segmenten van klanten jou het meest opbrengen, kan je daar initiatieven voor ondernemen.

**Slide 33:** Soorten clusters waarin je mensen kan onderverdelen. Tegenwoordig: Facebookprofielen. Heel vaak leidt een relatie in een sociaal netwerk tot dat je als persoon nauwer bij elkaar ligt qua profilering.

**Slide 34:** Komt nog uitgebreider aan bod, maar is een overzicht van clusteringtechnieken die bestaan.

**Slide 36:** Customer lifetime value modeling: de info die je (nodig) hebt om aan churn prediction/response modeling te gaan doen, daarvoor heb je CLV nodig. Het gaat erom om een model te bouwen dat kan inschatten/berekenen hoeveel een bepaalde klant jou waard is over een bepaalde tijdsperiode. We maken een som van revenue-kosten op een bepaald ogenblik t (of in interval t) maal s: survival rate: probability that customer is still alive at time t. Je gaat die verdisconteren: rekening mee houden dat geld in de toekomst een andere waarde heeft dan op dit moment. Time horizon: in telco vaak 2-3 jaar. Per maand/kwartaal doe je die berekening en krijg je een inschatting van de waarde van jouw klanten.

**Slide 37:** In veel bedrijven zit je met verschillende soorten producten die gebruikt worden door jouw klanten. Het kan zijn dat je modellen bouwt die geen rekening houden met bv X-selling of dat je geen rekening houdt met cannibalisme: als je een bepaald product meer verkoopt aan een klant, dat zal leiden tot een mindere verkoop van een ander product. → Veel complexere CLV dus!

**Slide 38:** Let op, de bedragen langs rechts zijn nog niet gedisconteerd, enkel het onderstaande cijfer is verdisconteerd!

**Slide 40:** SNA: trend binnen bedrijven gegeven het belang van sociale netwerken om de data/info te gaan gebruiken voor allerlei marketinginitiatieven. Dus niet enkel op social networks marketing doen, maar ook in andere marketinginitiatieven gebruik maken van social networks.

Hoe in telco een sociaal netwerk? Call record details: CRD: wordt gebruikt om een sociaal netwerk van klanten op te stellen.

**Slide 41:** Hoe social network gebruiken binnen een bedrijf. Effici¨entie en werking van bedrijf analyseren en verbeteren. Ingekleurd op basis van departement en e-mails die verstuurd zijn. Grijze nodes: externe experten, gele: consultants. Communicatieflow is dus niet zo fantastisch en als die verloopt, is dat vaak via consultants. Ze zijn gaan nadenken of dit wel ok is en op basis van deze analyse is men gaan nadenken of de mensen bij verschillende departementen, of ze die fysiek zouden moeten samenzetten zodat ze veel effici¨enter met elkaar kunnen communiceren.

**Slide 43:** Voorstelling sociale netwerken: sociogram: visualisatie van sociaal netwerk. Bolletjes zijn nodes, streepjes zijn links. Je kan het ook in matrixvorm voorstellen: matrixrepresentatie van een sociaal netwerk.

**Slide 44:** Als de nodes mensen zijn, spreken we van een echt sociaal netwerk.

**Slide 45:**

* Churn prediction: effect uitspelen dat klanten mekaar be¨ınvloeden. Er is een verhoogd risico dat jij zal churnen indien heel veel van jouw vrienden ook gechurned hebben/zijn.
* Sociaal netwerk van banken: stel dat A failliet gaat, wat zijn de implicaties, wat zijn de sterke connecties met andere mensen en wat is het risico dat andere banken ook failliet gaan?
* Fraude: netwerk van bedrijven: kijken naar welke werknemers overstappen naar welk anders bedrijf.

Je hebt heel sterke influencers: als je 1 persoon kan overtuigen, vaak ook veel andere mensen.

**Slide 47:** We gaan in rekening brengen dat als jij als klant geconnecteerd bent aan klanten die al gechurned hebben/waarvan de kans groot is dat zij gaan churnen, is die kans bij jou ook groter. Er is een verhoogd risico op churning als mensen in jouw nabijheid churnen. We kunnen dat ook gaan gebruiken.

**Eerste orde:** Stel dat Victor en Sophie gechurned zijn, gegeven de connectie met Bart, kunnen we gaan zeggen dat de kans voor Bart groter zal worden. We kijken naar de nodes die geconnecteerd zijn met gechurnde nodes.

**Tweede orde:** Vrienden van vrienden. Kans dat Bart churned is afhankelijk van de kans dat een van die tweede-orde nodes churned.

Op die manier kunnen we voor Bart een probabiliteit berekenen die rekening houdt met statische variabelen maar ook met zijn sociaal netwerk.

**Slide 49:** Sentimentanalyse: kijken wat de klanten op sociale media over onze diensten zeggen en dat analyseren we met het oog op het verbeteren van onze producten/om klanten op een betere manier bedienen.

**Slide 50:** Businessmodel van Facebook is voor 100% gebouwd op advertising. Heel eenvoudige interface die ongelooflijk veel geld opbrengt. Laat toe om bv clicks naar jouw website te genereren, je kan ook zelf een pagina maken op Facebook en zo likes verzamelen. Je kan als adverteerder gaan bieden om bepaalde segmenten te gaan targetten (zowel geografisch als op basis van connecties tussen mensen).

**Slide 51:** Doelstellingen zetten: wat wil je?

**Slide 52:** Dan selecteren welke audience je wilt. Daar is Facebook veel sterker in dan andere bedrijven. Facebook kan veel gedetailleerder targetten omdat ze zoveel informatie hebben over ons. Zelfs als het profiel niet aan iets bepaald voldoet, kan je ingeven dat je samenhangt aan een ander bedrijf/andere pagina.

**Slide 53:** We willen inschatten hoe goed/slecht ons product gepercipi¨eerd wordt door de klant. Heel belangrijk om te verstaan wat klanten van jouw product vinden zodat je het kan aanpassen. Je moet weten hoe goed jouw product in de markt staat en daar hebben producten het vaak moeilijk mee.

**Slide 54:** Foutief gebruik maken van sociale media: NYPD: hoopten te kunnen inschatten of de dienstverlening positief werd gepercipi¨eerd. Een massa aan foto’s van situaties waar NYPD-agenten in niet zo’n goed daglicht werden gesteld.

**Slide 55:** Zoeken naar hoe goed klanten jouw product/dienst vinden. Twitter laat toe om alle tweets over jouw product uit Twitter te halen om zo die data te analyseren. Je kan dan tekstanalyse doen waarbij je elk woord in een tweet gaat classificeren als positief/negatief/neutraal, daarna ga je een analyse over die tweet doen en als je dan veel tweets beschouwt, kan je zo een globale analyse doen.

**Slide 58:** Je mag als bedrijf heel veel data gebruiken, maar altijd aandacht hebben op privacy: wees attent voor de privacy van jouw klanten.

**Chapter 20**

# Les 17: 27/04/2015

## 20.1 Slides: 6A.BusinessIntelligenceandDataAnalytics

**Slide 2:** We gaan kijken naar business intelligence en data analytics. Vandaag: BI in het algemeen en Data Warehousing. 2 volgende sessies: DA.

**Slide 4:** BI: IS’en, er bestaan allerlei verschillende soorten. We gaan praten over die IS’en die de beslissingnemers in een organisatie ondersteunen. Decision support systems etc gaan wij bespreken (hun technieken). BI is de brede term. Deze technieken laten ons toe om goede beslissingen te nemen. Bedrijven worden geconfronteerd met problemen: bedrijven krijgen massa’s data. Ze worden geconfronteerd met een situatie waar de besislissingsnemers overrompeld worden met massa’s data. We willen systemen opzetten die kunnen omgaan met die al die data. Er wordt nu veel data beschikbaar en de markt verandert heel snel. Beslissingen moeten ook steeds sneller worden genomen omwille van veranderde marktomstandigheden, dynamische omgevingen waarin bedrijven moeten werken. Dat leidt ertoe dat we systemen nodig hebben die ons kunnen helpen. We gaan het traditionele verhaal bekijken: data verwerken, daar info van maken en dat gebruiken om er kennis uit te halen om dan actie te kunnen ondernemen.

**Slide 5:** Als we praten over het organiseren van die data in functie van het nemen van beslissingen, spreken we over BI. Soort van definitie op de slide: BI is een proces binnen een organisatie waarbij het gaat om data op zo’n manier te structureren dat je die kan uitbuiten om beslissingen te nemen. Die data moet op zeer gestructureerde manier worden georganiseerd. We moeten de data dus organiseren en analyseren om er beslissingen uit te kunnen halen.

**Slide 6:** We praten hier over de 2 hoogste niveaus in de piramide: management en executive level → beslissingen zijn hier het talrijkst en het moeilijkst. Op operationeel level is het vaak niet moeilijk om beslissingen te nemen (bijna triviaal). Bovenaan staat de definitie van BI: data gebruiken om beslissingen te ondersteunen. Heel wat technieken/tools die binnen het domein van BI vallen. Je zou daar veel meer onder kunnen positioneren, maar nu gaan we gewoon de meest interessante/gebruikte technieken toelichten. Deze 4 zijn opgelijst. Data warehousing en OLAP hangen heel sterk samen.

**Slide 7:** Beslissingen nemen: kunnen antwoorden op vragen die beantwoord moeten worden door beslissingsnemers in de organisatie. Er staan hier 2 soorten vragen gepositioneerd: 2 niveau’s van moeilijkheid. Degene in de box zijn ”makkelijke” vragen: verificatie-gebaseerde vragen. Deze kunnen we beantwoorden met simpele BI-technieken.

**Slide 8:** BI is zeer populair, komt heel hard op. Heel veel tools en technieken worden momenteel in grote mate gecommercialiseerd en verkocht omdat de voornaamste driving factor data is. Bedrijven weten vandaag de dag veel minder hoe om te gaan met data. Er zijn bedrijven die dat wel weten en op die manier waarde cre¨eren voor hun bedrijf. De technologie die we vandaag hebben omdata te analyseren, zorgt voor een push naar bedrijven toe om die data te gebruiken.

**Slide 9:** 2 soorten moeilijkheden:

**Verification-based techniques:** Kunnen beantwoord worden aan de hand van simpele BI-technieken. Je gaat iets verifiëren, niet berekenen, gewoon opzoeken/verifiëren.

**Discovery-based techniques:** Moeilijkere technieken: daar valt data mining onder. In deze soort technieken moet je nog effectief op zoek gaan naar nieuwe patronen. In de eerste set technieken weet je waar je naar opzoek bent. In deze weet je dat totaal niet. Bij discovery ga je zo zoek naar ongekende patronen, zaken die je nog niet wist. Data mining: gaan op automatische manier in data op zoek naar ongekende patronen.

**Slide 10:** Verificatietechnieken bestonden in principe al voor de PC: data verzamelen om te zien wat de situatie (financieel, operationeel, markt) is van je bedrijf → rapporten bouwen, score cards maken die aangeven hoe goed/slecht jouw bedrijf het doet. We noemen deze apps ook analytic/BI applications: laten toe een overzicht te krijgen van je organisatie aan de hand van de gegeven technieken. Het gaat erom om op een relatief geaggregeerd niveau de situatie van je bedrijf te kunnen rapporteren. De informatie is dus al beschikbaar, je moet ze alleen samenbrengen, aggregeren in een rapport zodat de beslissing nemers hun beslissingen kunnen gronden. Statistieken presenteren aan mensen die er dan mee aan de slag kunnen.

**Slide 11:** BSC: basisreferentiemodel voor alles wat te maken heeft met enterprise reporting. Gaat de performantie van heel het bedrijf in 1 picture trachten te vatten. Heel interessant hierbij is dat traditionele reporting gaat focussen op financiële info (winst,...). K&N vinden de financiële aspecten onvoldoende, er is meer nodig. Ze hebben er dimensies aan toegevoegd: financials toevoegen aan BSC, maar we moeten dat aanvullen. Er zijn 3 additionele dimensies die nodig zijn om de performantie van de organisatie in kaart te brengen:

**Customer:** Bv klachten, hoeveel support hebben klanten nodig? Wordt steeds belangrijker!

**Internal Business Processes:** Alles wat te maken heeft met operationele dimensies. Bv order processing time: ben je in staat om je orders tijdig te verwerken? Wat is de throughput time van orders en voldoet dat aan de verwachtingen?

**Learning and Growth:** Alles wat te maken heeft met het hebben van goed getraind personeel. Kan je als organisatie de skills uitbouwen die je nodig hebt om je services/producten aan de man te brengen?

Hoe gaat het effectief in zijn werk: 2 fasen bij het opstellen van BSC:

1. Definiëren van metrieken (die we meten). Je gaat voor elk van die dimensies een 8-10 indicatoren definiëren en je zet een target value voor elke indicator.
2. Meten: data opvragen, databases raadplegen en nagaan of die targets effectief behaald worden. Je gaat de targets verifiëren met de feitelijke waarden. Dat doe je voor elke indicator en op basis van een studie van target vs werkelijk ga je proberen te verbeteren. Zo maatregelen/nieuwe beslissingen nemen om die targets wel te halen.

Overall view in 1 beeld van de performantie dus.

**Slide 12:** Resultaten van een survey naar welke indicatoren voorkomen in zo’n BSC. KPI’s: worden vaak opgenomen in BSC’s/digital dashboards.

**Slide 13:** Veel varianten van BSC, nu ”moet” je verder gaan dan BSC die specifieker zijn voor specifiekere dimensies (bv dashboard voor alleen dingen van klanten). Je gaat ook hier targets zetten voor bepaalde KPI’s en die dan opvolgen. De laatste 5-10 jaar is er een enorme push geweest om dat alles heel mooi op te nemen. Managers willen zo’n systemen, waar ze in 1 oogopslag kunnen weten hoe goed het vandaag gaat met de organisatie. Digital dashboards: verfijningen van de originele BSC.

**Slide 14:** CPM: hier is er een meer functioneel georiënteerde view. Je gaat hier kijken naar de verschillende functionele domeinen en maakt voor elk van die systemen een functionele analyse. Je kan dashboards bouwen, aan CPM gaan doen voor specifieke functionele domeinen. CPM is meer functioneel georiënteerd.

**Slide 15:** Financial view waarbij je indicators krijgt (traffic lights/meters, grafieken/tabellen) → visueel duidelijk maken of er afwijkingen zijn tussen targets en indicatoren.

**Slide 16:** BAM: real-time corporate performance management. Het gaat ’m voornamelijk over het operationele niveau, het managen van business processes. Is verschillend van CPM in zijn geheel, is vaak niet real-time, bij BAM wel. Je gaat hier een continue connectie hebben met de operationele databases om de juiste data te kunnen queryen om die metrieken te kunnen opvolgen. Veel organisaties wensen dit uiteraard.

Als mensen spreken over BI, gaat het in 99% van de gevallen over wat we tot nu toe al besproken hebben. In heel veel gevallen stopt het daar. BI is veel meer dan dat. Er is een heel groot verschil tussen hoe wij naar BI kijken en hoe BI effectief wordt toegepast en gerealiseerd in de praktijk.

Basisverschillen tussen de termen kennen, maar niet in detail!

**Slide 17:** Verificatie-techniek: OLAP: een van de belangrijkste applicaties van data warehousing. Het gaat er hier om dat je een systeem bouwt die bovenop een database wordt geplaatst die toelaat om de data interactief te gaan queryen. In tegenstelling tot die dashboards die we tot nu toe zagen (eerst definiëren van metrieken en die dan opvolgen), gaan we hier interactieve, dynamische queries hebben, niet op voorhand gedefinieerd. We gaan hier software hebben waarmee we kunnen beslissen wat we juist zoeken om bepaalde inzichten te verwerven die nog niet gecheckt zijn. Je gaat de databases queryen op basis van een ad-hoc query: je komt met een vraag, je wil die laten beantwoorden, dat kan dan via een OLAP-cube die je de database laten queryen.

**Slide 18:** De moeilijkere variant van BI. Je gaat hier niet op zoek naar simpele feiten/rapporten aanvullen, maar je gaat op zoek naar nieuwe patronen. DM in zijn algemeenheid gaat over het ondersteunen van beslissingen door nieuwe patronen te ontdekken, nieuwe modellen te bouwen (churn prediction! → Model bouwen om churn te voorspellen, want is ongekend patroon/model).

**Slide 19:** DM laat toe om te antwoorden op de complexere vragen die hier omkaderd zijn.

**Slide 20:** Startpunt van alles wat met BI te maken heeft. Data warehousing: organiseren van je data in een specifieke database die gebouwd is om die BI-tools op los te laten. We hebben ER en SQL gezien, maar da’s operationeel/ontwerpen. Nu gaan we het hebben over het opstellen van analytische DB’s → data warehouses: specifieke databases met specifieke structuur/ontwerp die gebouwd zijn om BI apps op los te laten.

**Slide 22:** Data warehousing is de start van alles in het domein van BI. Alvorens je de mooie dashboards kan bouwen, moet je weten waar je de data kan vinden. Vaak ziet het operationeel systeem eruit zoals links: gefragmenteerd, heel veel verschillende databases. Als je dat moet gaan gebruiken om die dashboards te bouwen/data mining te doen, kom je nergens, is veel te complex. Die data is niet geaggregeerd. Met data warehousing gaan we de data samenbrengen (dus de blauwe pijl) in 1 database. Vanuit die ene gecentraliseerde database kunnen we dan de tools gebruiken. Dit is vaak de moeilijkste stap: het organiseren van de organisationele data is een gigantisch complexe opdracht.

**Slide 23:** De data kijkt als input op je systeem. Data halen vanuit je operationele databases, maar soms ook vanuit externe bronnen. We gaan de data warehouse zo organiseren dat die centraal georganiseerd is rond subjecten in je organisatie. In plaats van applicaties/systemen die transactiegericht zijn (bv operationeel systeem: productieproces → ontworpen volgens het principe dat je elke stap kan opvolgen), gaat alles subjectgeorganiseerd zijn.

**Slide 24:** Belangrijk te weten wat het verschil is tussen OLAP (datawarehouse database) en OLTP. Moesten OLTP databases perfect bruikbaar zijn voor BIapps, was OLAP niet nodig, maar OLTP heeft kenmerken die ervoor zorgen dat ze niet goed bruikbaar zijn voor BI. OLTP is ontworpen voor transaction processing: continu nieuwe data wegschrijven naar de DB. Continu interageren om de database up te daten. Ook zorgen voor concurrency control want de data wordt door meerdere personen gelijktijdig benaderd. Nieuwe sales worden bv continu toegevoegd aan de database. Je moet ermee kunnen omgaan dat meerdere users data toevoegen/updaten. Ook kunnen omgaan met failures van de database (ook nodig voor OLAP). Belangrijkste verschil zit in de 2 bovenste puntjes: transaction en concurrency. Om dat goed te kunnen doen heb je normalisatie nodig. Dat is de reden waarom operationele databases zo ver mogelijk genormaliseerd zullen zijn. Dat zorgt ervoor dat je geen updatefouten maakt, geen redundantie hebt (want dat is het doel van normalisatie). Je gaat aan de hand van het verhinderen van redundantie ervoor zorgen dat je data niet gedupliceerd hebt in je database. Als je data updatet, zorg je dat dat consistent gebeurt, je geen fouten maakt. Normalisatie maakt ook dat je data heel goed kan updaten. Belangrijk: OLTP databases zijn volledig genormaliseerd, maar als we op zo’n database analytische queries loslaten (typische BI-queries zijn heel grote queries), moet je een heel goede performantie hebben van je database om complexe queries te kunnen uitvoeren. Uitvoeren van zo’n query op zo’n database is zeer weinig performant want alle data is in verschillende tabellen opgeslagen. Je moet die allemaal gaan joinen om 1 view op je database te kunnen krijgen. Die JOIN’s zijn dus heel kostelijk, heel duur in termen van rekenkracht. Daarom zijn OLTP databases niet geschikt voor BI. We kiezen dus voor niet-genormaliseerde DB’s voor performantie.

**Slide 25:** Specifiek gebouwd voor BI en het uitvoeren van complexe queries. Geen/weinig normalisatie om JOIN’s te voorkomen om zo performantere/complexere queries uit te kunnen voeren.

**Slide 26:** Beetje verloren, maar soort van samenvatting: samenbrengen van externe data. Gaat dus overall view geven van alle data die relevant is voor de organisatie. Is cross-functional, niet voor 1 enkele functie, maar gaat alle data in de organisatie bevatten.

**Slide 27:** Kernkarakteristieken (deze 2 zijn de belangrijskte, er komen er nog!):

* Time-variant: je moet telkens het verschil kunnen leggen tussen OLTP en OLAP. OLAP databases zijn time-variant. OLTP databases zijn time invariant. Als je naar een OLTP database kijkt, is die time invariant omdat op elk moment dat je naar zo’n database kijkt, je de recentste data krijgt. Er is maar 1 waarheid en dat is de huidige waarde van de records in je database. Wanneer je er ook naar kijkt, je gaat altijd de laatste versie van je data hebben. Van OLAP: time variant: hangt ervan af welke tijdsspecificatie je meegeeft om ernaar te kijken. Je krijgt andere versies van de data records. Eigenlijk is een data warehouse een soort verzameling van historische snapshots van jouw operationele database. Een data warehouse kan aan de hand van tijd gequeried worden op verschillende manieren en zo kan je een andere view krijgen. In een operationele database kan je nooit op 1 bepaald moment meerdere soorten views krijgen, altijd maar 1 versie van de waarheid. In OLTP database: time-invariant: altijd maar 1 versie van de waarheid. In data warehouse is dat niet zo. Daarom is er ook non-volatility.
* Non-volatile: in data warehouse databases ga je nooit data updaten/verwijderen. Het enige wat je gaat doen is data toevoegen. Daarom is een data warehouse database non-volatile. Het gaat wel wijzigen, maar je gaat alleen maar data toevoegen, nooit wijzigen/verwijderen. Dat is niet het geval van OLTP-systeem, daar ga je continu je data updaten.

Die 2 verschillen zijn al het belangrijkste om een data warehouse te snappen.

**Slide 28:** Karakteristieken, maar je moet dit niet vanbuiten kennen. Examenvragen hierover gaan polsen naar inzicht, maar je moet nooit definities aframmelen/kijken welke karakteristieken moeten aangevuld worden ofzo. Kijken naar boodschap achter termen.

Data warehousing is integrated: ge¨ıntegreerde database. Bij operationele databases zijn ze meestal niet ge¨ıntegreerd.

Time (in)dependent == time (in)variant.

Subject/oriented/transation oriented: subject: georganiseerd rond de belangrijkste objecten, terwijl bij operationeel systeem: alles transactie-oriented.

**Slide 30:** We hebben gepraat over het verschil tussen OLTP en OLAP. Nu gaan we specifiek ingaan op hoe je zo’n OLAP database ontwerpt en hoe je die gaat gebruiken. Het ontwerp/design van een data warehouse: het basismodel/fundamentele basismodel is het multidimensionaal model. Dit model (OLAP-cube) is verre van perfect maar het is het model dat typisch gebruikt wordt om over een data warehoues te praten of dat weer te geven/tekenen/tonen. Dit is een N-dimensional cube, in dit geval 3-dimensioneel. Dat bevat een goede representatie van wat een data warehouse is, maar een data warehouse bestaat typisch over honderden dimensies en niet gewoon 3. Als je een data warehouse wil gaan modelleren/tekenen, lukt dat uiteraard niet meer. Je kan dit gebruiken om op een hoog niveau te praten over een data warehouse, concreet uitleggen hoe en data warehouse eruitziet als je beseft dat er meer dan 3 dimensies zijn. Het is zo dan ook makkelijker uit te leggen hoe zo’n data warehouse eruitziet/ontworpen kan worden. 2 fundamentele concepten: dimensie en measure values/facts. Alles in een data warehouse gaat om dimensies en facts. Elk model dat we nog gaan bekijken, gaat een definitie/organisatie geven van dimensies en feiten. Voor elk van die dimensies ga je observaties krijgen. Je krijgt een getal: stel dat het een data warehouse is die het verkoopvolume bijhoudt voor de verschillende kwartalen in verschillende vestigingen. Zo krijgen we een bepaalde value, een bepaald fact.

**Slide 31:** Je moet beseffen dat die facts relatief makkelijk begrijpbaar zijn. Het zijn gewoon waarden. Het concept van een dimensie is moeilijker omdat aan een dimensie steeds een hiërarchie vasthangt en die is fundamenteel over data warehouses en data warehouse-querying. Dat gaat allemaal uit van dat een dimensie van een data warehouse hiërarchisch gestructureerd is. Je hebt verschillende niveaus waarop je naar die dimensie kan kijken: je kan kijken naar alles, per categorie of per product. Andere dimensies zullen ook telkens een hiërarchie hebben. Die hiërarchieën laten toe om te doen waar data warehouses voor gebouwd zijn: queryen van data voor verschillende hiërarchische levels van dimensies. De hiërarchieën zijn dus fundamenteel.

De facts zijn in principe simpel, maar als je dat samenbrengt met de hiërarchieën, moet je gaan opletten.

**Slide 32:** Als je die (facts) wilt gaan aggregeren, moet je zien dat je de feiten uit elkaar kan trekken of samen kan brengen (verkopen volgens tijd bv). De facts moeten summarisable zijn. Er zijn facts die maar half-optelbaar zijn (semi-additive): voorraadniveau is een fact, maar het is niet optelbaar. Het voorraadniveau in week 1 en week 2 samen is niet de som van de 2 voorraadniveau ’s. Sommige metrieken/facts zijn niet-optelbaar en daar moeten we dus gebruik maken van eventueel andere functies, zoals bv het gemiddelde. Bij andere metrieken/feiten is het veel moeilijker om de correcte (dis)aggregatiefunctie te definiëren.

Het multidimensionaal model: dimensies en feiten: dimensies: hiërarchieën nodig en die feiten moeten summarisable zijn.

**Slide 33:** We hebben ER-modellering gezien, dat gaat erom om op een conceptueel niveau een database te definiëren: het conceptueel ontwerp van een database. Je zou ER-modellering kunnen toepassen op het ontwerp van data warehouses. dat is een probleem want fundamenteel kan/heeft een ER-model niet de notie van een hiërarchie. Wel de notie van objecten (dus dimensies zijn mogelijk), maar de hiërarchieën zijn niet mogelijk. Er is geen manier om met hiërarchieën om te gaan in ER. Je wil dus een conceptuele taal die beter is voor dimensies, hiërarchieën en feiten, die dat op een natuurlijke manier laat modelleren. Er is al veel onderzoek rond gedaan, maar er is zo geen taal.

De modellen die gebruikt worden zijn o.a. de OLAP-cube (maar 100 dimensies weergeven is niet mogelijk). Daarom gaan veel mensen die met data warehouses in de praktijk werken meteen werken op het logische niveau (niveau dieper → conceptueel, logisch, fysisch). Je moet beseffen dat er bij data warehouses geen goede manier is om op het conceptueel niveau te modelleren, alleen op een logisch niveau.

**Slide 34 & 35:** Op logisch niveau heb je 3 varianten van data warehouses:

* Relationele OLAP’s (ROLAP): data opslaan in relationele modellen.
* MOLAP: ligt veel meer in lijn met het multidimensionaal model, laat toe om typisch veel efficiënter te queryen dan in ROLAP maar is moeilijker te implementeren want wordt maar door ”minder mensen verstaan dan ROLAP want deze wordt al vaak gebruikt in relationele systemen, dus neiging om ROLAP gewoon te gaan herbruiken.
* Hybride: niet te kennen.

**Slide 36 & 37:** *De* manier om een data warehouse te tekenen op papier. Dit model bevat alle eigenschappen die je moet hebben om een data warehouse te kunnen modelleren. Het gaat om facts en dimensies die je gaat structureren. Facts/measurements komen in de centrale tabel (facts table). Dat is de tabel die onze facts (in het roze) zal bevatten, de values ervan. Ook nog andere datavelden: sleutels die verwijzen naar de dimensietabellen (die aan de kant). Heel weinig feitentabellen: 1/heel weinig en heel veel dimensietabellen. Daar wordt naar verwezen aan de hand van een sleutel in de dimensietabel die verwijst naar de feitentabel (denk ik?). De dimensietabel laat dan toe om extra info te vinden over bv een sales fact.

Merk op: die dimensietabellen zijn niet genormaliseerd! Die dimensietabellen rond de feitentabel zijn niet genormaliseerd. Als die dus helemaal niet genormaliseerd zijn, noemen we dat een sterschema.

**Slide 38:** Dimensional keys verwijzen naar dimensietabellen en deze bevatten dimensieattributen. Die dimensional keys vormen samen een composed key voor elk (sales) facts. De samenstelling van die sleutels vormt de composed key die refereert naar een sales fact. Dat is de manier om op een logisch niveau een data warehouse te vormen.

**Slide 39:** Je gaat een facts table hebben met sleutels die verwijzen naar andere tabellen in de database. Op die manier spreek je over een relationeel model van de warehouse (relationeel want via sleutels verwijzen naar de tabellen).

**Slide 40 & 41:** Sterschema is de basisvariant: het is een datawarehouse waarbij de dimensietabellen niet genormaliseerd zijn: per dimensie juist 1 tabel. Heel veel van die tabel kunnen soms heel groot worden daardoor, met heel veel attributen, met dus ook een kans dat als je records toevoegt, je in de problemen komt (vraagt veel tijd, fouten ontstaan,...) → snowflake schema: normaliseert de dimensietabellen tot op een bepaald niveau.

Store en product: 2 dimensietabellen die tot op een bepaald niveau genormaliseerd zijn: niet alle info over city en state: citykey en statekey. Op die manier gaan we iet of wat redundantie van data verhinderen. Stel dat we al die info toch in de store database zouden opslagen, zouden we voor elke store in een stad dezelfde data hebben.

**Slide 42:** Bij een sterschema, als je dat gaat gebruiken, ga je 1 JOIN doen (product en sales), bij het snowflake design ga je meerdere JOIN’s moeten doen, kost meer tijd en rekenkracht en dus zal jouw query minder performant zijn. Dit moet uiteraard afgewogen worden tegenover dataredundantie.

**Slide 43:** Data warehouse architecture: effectieve werking van data warehouse. We hebben verschillende niveaus waarover we moeten praten (opgelijst in de verschillende rechthoeken/tiers): verschillende stappen/tiers in data warehouse.

* Data sources: uit operationele en externe data data halen om warehouse te vullen.
* Front-end tier: reporting, data mining.

Hoe gaan we nu van data sources naar front-end tier? Soms hebben we 3 zaken nodig: back-end, data warehouse en OLAP. Het eerste is ETL: back-end: Extract, Transformation and Load (zie volgende slide).

**Slide 44:** Je hebt software nodig om data uit externe bronnen te verwerken.

**Slide 45:** Je hebt vaak te maken met integratieproblemen: er ontstaan vaak mismatches (eenheden,...), transformatieproblemen. Je gaat de verschillende velden scannen/verifiëren/nakijken en die gelijktrekken en ervoor zorgen dat de data gelijk weergegeven wordt. Je moet je inputdata gaan transformeren zodat je een uniform formaat krijgt.

**Slide 46:** Data staging (belangrijk!): realisatie van ETL gebeurt via data staging area: aparte database waarop de transformaties worden toegepast. De modificaties/aanpassingen aan de source data gebeurt dus in de data staging area. Zodra je tevreden bent met de transformaties die gebeurd zijn, ga je die in de EDW steken (zie ook **Slide 43**).

In essentie gaat het over de transformatie van data in een uniform formaat. In heel veel bedrijven wordt die overnight gedaan. Soms ook week per week (in het weekend), maar meer en meer gebeurt dit in kortere tijdspannes: om de x aantal uren updaten. In sommige gevallen gebeurt dit bijna real time (maar moet je niet onthouden).

**Slide 47:** Data warehouse database is geïntegreerd, alles samen, maar in sommige gevallen ga je de indruk wekken dat het niet zo is. En soms is het ook echt niet volledig geïntegreerd.

* Het generieke model staat links: enterprise-wide: heel vaak voorkomende architectuur van data warehouses.
* Soms kan je ervoor kiezen om dat deels te fragmenteren: independent data marts: als het te kostelijk/onmogelijk is om het samen te brengen. Omvatten een bepaald stuk van je organisatie. Is niet altijd ideaal. Soms wordt ervoor gekozen vanuit security redenen: bepaalde data is enkel zichtbaar voor een bepaald publiek en dan ook op fysisch niveau de data apart houden.
* Hybrid architecture: meer functioneel georiënteerd: soms is het interessant om voor een select publiek een virtuele aparte data mart te organiseren: een apart stukje van je data warehouse. Bv om end users data te laten queryen: je hebt niet altijd alle data nodig voor bepaalde queries.

**Slide 48:** Over OLAP server geen aparte slide. Je moet de databases kunnen benaderen. Dat kan ofwel rechtsreeks (OLAP tier overslaan), in sommige gevallen doe je dat beter niet omdat je tools niet in staat zijn om te kunnen communiceren: OLAP server legt connectie tussen allerhande tools en de database. In sommige gevallen is dat nodig, in andere gevallen kan de access direct zijn. Pijl onderaan: variant van architectuur waarbij je de data sources (extern en intern) rechtstreeks voedt aan jouw analytische tools, aan de front-end. Je gaat alle queries rechtstreeks op je relationele data laten werken. Dit kan enkel in uitzonderlijke gevallen. Als het mogelijk is, gaat het over een virtual data warehouse. Bv wanneer de organisatie volledig ondersteund wordt door 1 ERPsysteem die al BI/datawarehousing functionaliteit in zich heeft.

**Slide 49:** Verificatie vs discovery:

**Verificatie:** Het uitvoeren van ad-hoc queries op een data warehouse: OLAP. Er bestaan heel wat varianten van de queries die we gaan groeperen: OLAP operaties.

**Discovery georiënteerde tools:** In de volgende sessies.

**Slide 50:** Querying op data warehouse gebeurt meestal niet in SQL. Is beter in taal die overweg kan met hi¨erarchie¨en: MDX.

**Slide 51:** OLAP operaties: querying van data: afhankelijk van dimensies die je (dis)aggregeert spreken we over verschillende OLAP operaties/soorten queries. Er worden er 4 opgenoemd, maar er zijn er uiteraard veel meer.

**Slide 52:** Roll-up: gebruik maken van hiërarchieën van dimensies om data te (dis)aggregeren. Roll-up is aggregeren volgens 1 dimensie (in het voorbeeld: country).

**Slide 53:** Drill-down: tegengestelde van roll-up: 1 dimensie in hiërarchie specifiëren, disaggegreren: drill-down. In vb: die quarter moet eigenlijk month zijn. Je gaat specifiek dieper in 1 bepaalde dimensie. Heel belangrijk in roll-up en drill-down: je gaat kijken naar 1 dimensie.

**Slide 54:** Slice: je gaat de waarde van je dimensie uitbuiten: specifiek kijken naar 1 element/waarde van een specifieke dimensie. Je gaat slicen op basis van de stad Parijs in het vb: 1 dimensie specifiëren zodat je een dwarsdoorsnede van de kubus krijgt.

**Slide 55:** Meerdere dimensies specifiëren. Geen gebruik maken van hiërarchie (ook in slice niet). Je gaat 2 dimensies specifiëren aan de hand van ´e´en/meerdere waarden, maar het moeten wel minstens 2 dimensies zijn (anders spreek je over slice). Dice is dus de generieke variant van slice. Je krijgt als resultaat dan een subkubus.

**Slide 56:** Alles wat hier is uitgelegd is redelijk ”oude” kennis, er wordt heel sterk geïnnoveerd in het data warehousing domein. Tegenwoordig databases die alle data rechtstreeks in de computer kunnen inladen. Heel veel initiatieven/innovaties in dat domein.

**Chapter 21**

# Les 18: 04/05/2015

## 21.1 Slides: 6B.BusinessIntelligenceandDataAnalyticsstudentversion

**Vorige les:** BI en DW. Vandaag tweede les daarrond: eerste stuk rond data analysis.

Bereid de filmpjes voor tegen de volgende lessen! Stel vragen. → vanaf volgende maandag.

**Slide 2:** BI en DW zijn vorige les gecovered. Nu gaan we verder met een van de applicaties van DW’s. vorige keer: verification technieken en discovery (verificatie: wist op voorhand al wat je wou zoeken. Discovery: je moet nog ontdekken wat je wil zoeken). Vandaag gaan we kijken naar discovery-gebaseerde BI-technieken. We weten op voorhand nog niet wat we gaan zoeken, welke kennis we gaan ontdekken. Data analysis/data mining is daarom de algemene noemer voor verschillende soorten algoritmes die toelaten om in data op een automatische manier nieuwe patronen te gaan ontdekken. Dit is een gigantisch verschil met verificatie, daar wist je wat je wil weten. Hier is dat niet het geval, je gaat nieuwe kennis krijgen.

**Slide 3:** 5 karakteristieken over big data: bijzonder hot topic want omvat heel veel, komt heel hard op. Er wordt zoveel data geproduceerd binnen organisaties en men wil met big data er nu iets mee gaan doen. Verzamelnaam voor allerlei technieken. Je kan het ook veel nauwer gaan interpreteren, waar het dan gaat over grote volumes aan data. Er zijn heel weinig bedrijven met echte big data bezig. Als we over echte big data spreken, gaat het over TB’s per milliseconde. Big data gaat niet noodzakelijk over het volume, maar ook de snelheid waarmee die wordt geproduceerd. Het is een commerciële term om alles te beschrijven, maar nauwer gaat het om een subset van data analytics waar het gaat om gigantische volumes in korte tijd. We denken dan aan uitzonderlijke applicaties zoals bv in de geneesmiddelenindustrie of ruimtevaart (data doorsturen via satellieten). Als we ruimer gaan kijken zoals veel organisaties het invullen, gaat het om data analytics van beperkte, kleine datasets. We gaan technieken voorstellen die zeer goed werken op kleinere datasets.

Gigantische volumes aan data: tienduizenden jaren: biljoenen data. Later wordt

dat geproduceerd in een paar dagen. Stel dat we werken met DVD-schijven van 1GB, hoe groot is 5 exabytes als je dat voorstelt? → Voetbalveld: 50*x*100*x*50 (dus 50 m de hoogte in). De productie van data is in exponentiële groei. Vandaag de dag: elke 10 seconden 5 exabytes.

Dataopslag is vandaag de dag geen enkel probleem meer, het grote probleem zit hem in de vraag of we die data kunnen analyseren.

**Slide 4:** Heel veel data in heel verschillende domeinen. Het is niet gelimiteerd tot een bepaald stukje van de maatschappij. Er is heel veel vraag naar mensen die die data kunnen analyseren.

**Slide 7:** Gaan we die data kunnen analyseren? Het is angstaanjagend, want we weten het eigenlijk niet. Er is niemand die met zekerheid kan zeggen dat we in staat gaan blijven zijn om die data te analyseren. Het opslaan van de data is geen probleem, wel het analyseren.

**Slide 8:** Big data, zowel in de kleinste organisatie, tot de grootste organisaties ter wereld, de kansen de opportuniteiten die big data met zich meebrengen zijn ongelooflijk: self-driving cars, drones,... Zie TED video: computers zelf dingen laten leren. Men is gegaan van mensen die geneesmiddelen kunnen testen, die kijken of ze de juiste werking hebben. Een heel groot stuk daarvan wordt nu enkel met de computer gedaan. Ze kunnen zo selecteren welke moleculen het beste werken op een bepaalde ziekte. ⇒ in heel veel domeinen zien we het nut van data analytics.

**Slide 9:** Beschrijving van het domein waarover we spreken. We spreken over data mining/data analysis, maar in wetenschappelijke kringen wordt het als knowledge discovery in databases omschreven. Alles wat te maken heeft met data analysis valt daaronder.

**Slide 11:** Definitie: zie slide. Massive observational databases: het gaat over databases in het algemeen (dus een beetje sterk uitgedrukt). Het gaat erom dat je modellen en patronen wil ontdekken. Deze moeten voldoen aan een aantal vereisten, opgelijst op de slide.

* Valid: model/patroon vinden: moet ook gelden voor data die je nog niet gezien hebt. Google car: moet op elk moment een beslissing kunnen nemen: is dit een boom of een mens? Dat model is enkel geldig als je ook in de toekomst een goed onderscheid kan maken tussen een boom en een voetganger (moet niet alleen bomen/mensen kunnen herkennen vanuit zijn huidige kennis). → Toekomst!
* Novel: zo’n patroon/model moet nieuw zijn. De ontdekking van nieuwe patronen is de essentie, het gaat om dingen die je nog niet kent. Indien we ze wel kenden, ging het om verificatie.
* Useful: het gaat erover dat de patronen/modellen die je ontdekt, ook nuttig gebruikt moeten kunnen worden. Het is niet omdat een patroon valid en novel is, dat die ook useful is. Je moet zien dat mensen die die patronen/modellen krijgen, deze kunnen gebruiken. Google car: die technologie kan gebruik maken van die algoritmes om het onderscheid te maken tussen een voetganger en een boom.
* Understandable: minst belangrijke, maar mag je ook niet vergeten. Het moet begrijpbaar zijn voor mensen. Karakteristiek die in veel gevallen belangrijk is voor die modellen/patronen. Als mensen er gebruik van moeten maken, moet het begrijpbaar zijn. Als je in andere contexten dan de Google Car werkt, bv in bank: voorspelt of een bepaalde persoon zijn lening zal kunnen terugbetalen, dat moet begrijpbaar zijn voor de mensen in de bank. Als je niet weet waarom een algoritme/model aangeeft dat een klant zijn lening niet zal kunnen terugbetalen, zal je niets kunnen ondernemen (advies geven,...).

**Slide 12:** Data mining/data analysis is een onderdeel van een ruimer domein, te zien op de slide: maar 1 van de 6 stappen. We starten links. Stappen:

1. Eerst en vooral kijken naar de applicatie: vraag formuleren die we willen beantwoorden aan de hand van het data analysis model. Spreken we over churn prediction of bomen vs voetgangers? Samen met die vraag gaat ook het identificeren van source data: in je operationeel systeem zoeken naar die data die gebruikt kunnen worden om de vraag te gaan beantwoorden. Vaak is dat al een heel moeilijke stap.
2. Tweede stap: data selection: data sources die in data warehouse zouden kunnen zitten, selecteren. Beslissen welke data we gaan gebruiken. We kunnen die database dan een data mining mart noemen.
3. Cleaning: als je data al in een data warehouse zit, zal deze stap misschien niet zoveel tijd kosten (zie vorige les). In de praktijk is het vaak niet zo, dus moet je uw data gaan cleanen. Komt omdat data soms maar op een bepaald niveau wordt opgekuist. In sommige gevallen heb je nog extra stappen nodig om het te kunnen op te kuisen om het te kunnen gebruiken in een data mining applicatie.
4. Data transformation: eventueel! Na deze stap krijgen we een data set (Excel-file met kolommen). Elk van die kolommen kunnen we gaan transformeren (bv variabele inkomen, deze niet als absoluut cijfer meenemen, maar eerst categoriseren in laag, gemiddeld, hoog → = binning).
5. Data mining: qua tijdsinvestering misschien de minst belangrijke stap. Moment waarop je data mining algoritmes loslaat op de data om op een automatische manier modellen/patronen te ontdekken. → Heel geautomatiseerde stap waarbij heel weinig user intervention aan te pas komt. Je hebt de data gedefinieerd, beslist welke data je gaat toepassen en dan krijg je een model.
6. Interpretation and evaluation: KDD proces voltooit pas wanneer je die data ook gaat gebruiken: patronen evalueren (inschatten hoe accuraat ze zijn) en interpreteren (Snap ik ze? Zijn ze intuïtief correct? Stemt het overeen met wat ik verwacht?). Als je dat hebt gedaan, kan je beslissen om zo’n model te gaan gebruiken om bv een churn prediction model

te gaan gebruiken om potentiele churners op te sporen en actie te gaan ondernemen.

**Slide 13:** Data preprocessing: samenvoegen van selectie, cleaning and transformation. Wat doen met missing values? Als je een algoritme hebt dat automatisch patronen berekent en je zit met outliers, kunnen zij een sterke invloed hebben op jouw model/kan jouw algoritme daar misschien niet zo goed mee om. Regressie is exact hetzelfde: heel sterke outliers kunnen het regressiemodel heel sterk beïnvloeden, zodat de fit van je regressiemodel relatief slecht wordt. Definition target variable: hoe? Vaak niet moeilijk om ze te selecteren, maar om ze te operationaliseren is moeilijk! Je kan bv churn voorspellen, maar wat is de definitie van zo’n churn? Stel dat we spreken van passive churn: geen directe observatie van het churn-event. → Als organisatie gaan beslissen wanneer je iemand als churn gaat bestempelen.

**Slide 14:** Data mining bestaat uit 2 verschillende soorten technieken. Het grote verschil tussen beiden: bij predictief bevat de data een target variable: variable of interest. 1 variabele waarvoor we effectief een voorspelling willen maken. Churn prediction is daar een vb van: we weten welke variabele we willen voorspellen. → Supervised learning (je weet al welke variabele interessant is/je wil gaan voorspellen). Wanneer we die variabele niet hebben, spreken we over descriptieve methoden. Bij clustering weet je bv nog niet in welke segmenten/clusters je je klanten wil gaan opdelen.

**Slide 16:** Labeled data: dataset met een variable of interest. We werken met observaties en voor elke observatie werken we met 1 waarde voor de variable of interest: label. Dit is de reden waarom we spreken over predictieve data mining. Als we die variable of interest niet zouden hebben, zouden we kunnen gaan clusteren,... → gebruiken voor andere doeleinden, maar niet voor classificatie. We hebben onze data set, die noemen we de training set: gebruiken we om het model te leren. Algoritme loslaten op de data om daar een model te krijgen. We gaan daar bv decision trees op loslaten. Zo’n technieken kunnen modellen leren. We laten het learning algorithm los op de data, leren een model en krijgen een model. Je kan dat model gaan toepassen op nieuwe observaties. Unobserved: je kent de klassevariabele nog niet. Je kan het bekomen model dan nemen en een classificatie gaan toekennen aan de observaties waarvan je dat nog niet kende.

**Slide 17:** Predictief: opdelen in 2 categorieën: regressie en classificatie. Grote verschil: target variable. Als die target variable continu is: regressie. Als die discreet is: classificatie. Aan de hand van regressie voorspelling gaan doen voor nieuwe variabele.

**Slide 18:** Classificatie: wordt in de meest praktische toepassingen gebruikt. Variable of interest zal heel vaak discreet zijn. Technieken zijn opgelijst, maar niet exhaustief. Degenen die getoond zijn, zijn belangrijk voor ons. We gaan enkel naar de twee eersten kijken. Logistic regression: speciale vorm van regressie (je zou kunnen denken dat het thuishoort bij regressie). Je krijgt een score en aan de hand van van het teken kan je zien bij welke klasse je ligt.

Neemt vormen aan tussen 0 en 1 of -1 en 1: zal gebruikt worden voor classificatie, omdat je zo kan gaan voorspellen tot welke klasse het gaat behoren. Hier: als negatief: behoort tot klasse -1. bij -0.8 ben je natuurlijk zekerder over het behoren tot -1 dan bij -0.2. Zie Figuur 21.1 Neurale netwerken zijn zowel voor classificatie als regressie bruikbaar. Neurale netwerken en decision trees gedetailleerd kennen!

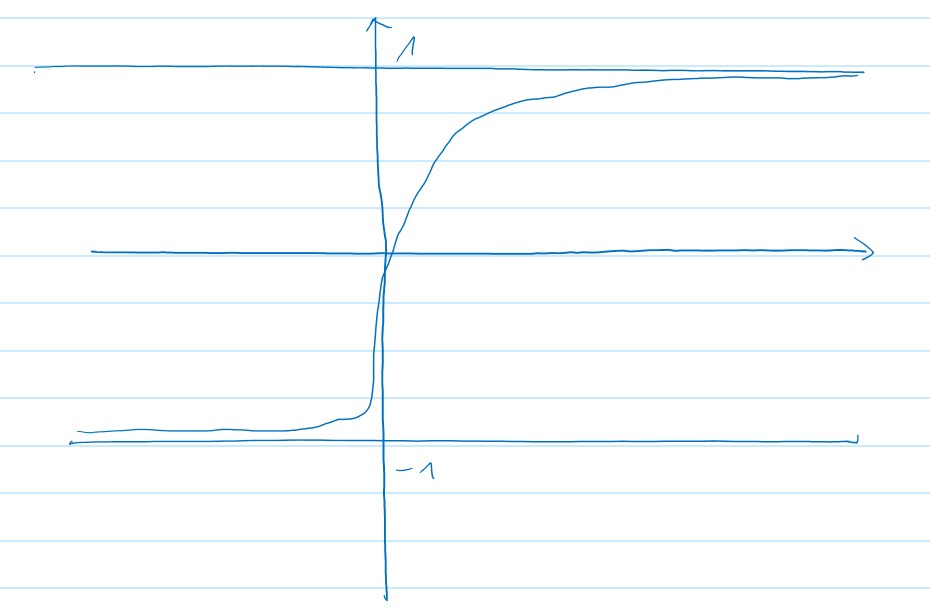


Figure 21.1: Logistic regression.

**Slide 19:** Observaties die worden gekarakteriseerd door een set van variabelen. We hebben onafhankelijke variabelen en 1 variable of interest: discrete variable of interest die een discrete waarde kan aannemen.

kNN is de meest triviale classificatietechniek. We hebben een dataset en nieuwe observaties waarvoor we de waarden van de variable of interest niet kennen, wel van de andere waarden. We gaan dan een predictie maken voor de variable of interest. Als we dieren classificeren en de variable of interest is ”eend” of ”geen eend”, gaan we kijken naar alle andere dieren en dan kijken welk dier het meest lijkt op wat we nu hebben, maar waarvan we nog niet weten of het een eend is of niet en we gaan voorspellen of het een eend is of niet. We gaan afstanden berekenen: kijken hoe sterk de test record lijkt op de dieren in de training records. Distance berekenen op basis van de variabele. Stel: de twee blauwe pijlen zijn degenen die er het kortste bijliggen, dus test record is ook een eend.

Het is simpel omdat je geen model hebt, al jouw training instances zijn jouw model. Je gaat gewoon al jouw training instances nemen en op basis van je distance metriek ga je dan classificeren.

**Slide 20:** Je hebt ook de value of k nodig: k is de essentiële parameter: hoeveel van de buren ga je in rekening brengen om te classificeren? In het vb op de slide is k 3: alle drie positief, dus unknown record zal ook positief zijn. Als er 2 plusjes stonden en 1 min, zou het ook positief zijn. → Op basis van majority prediction kijken wat de gezochte waarde zal zijn.

**Slide 21:** Belangrijk/moeilijk: tunen van parameter k: kiezen van de waarde van de parameter. Op **Slide 21** is k veel te groot genomen: in de buurt van x liggen alleen maar positieve observaties, maar doordat k = 20, ga je x op negatief zetten. Je gaat observaties in aanmerking laten komen die te ver afliggen van jouw nieuwe label. Je kan het soms ook te klein kiezen. Als k te klein is, kan een noise point leiden tot een foutieve classificatie.

**Slide 22:** Voordelen:

⊕ Geen tijd spenderen aan het (aan)leren van een model. Het leren van je model kost 0 seconden tijd, je moet alleen de data opslaan.

⊕ Zo’n ”model”/techniek kan online, in real-time geüpdatet worden. Omdat je geen model hebt, kan je in real-time jouw ”model” aanpassen en nieuwe instances toevoegen.

⊕ Je hebt maar 2 variabelen die je moet kiezen: k en distance metriek. Als we alle technieken in data mining gaan vergelijken, zijn dat weinig parameters (die dus ook moeilijker in te stellen zijn).

Nadelen:

Het feit dat het model trainen geen tijd kost, is meteen ook het grootste nadeel: classificeren van nieuwe instances kost heel veel tijd. Je moet de afstanden opnieuw berekenen tot de nearest neighbours. Als je met veel instances zit, moet je *alle* afstanden opnieuw berekenen. → Belangrijkste nadeel. Gebalanceerde data sets: stel dat je 100 observaties hebt. Balanced data set wilt zeggen dat uw variable of interest ongeveer 50/50 verdeeld is. Unbalanced is dus het omgekeerde: je gaat maar een klein aantal voorkomens van 1 mogelijkheid hebben en heel veel voorkomens van de andere. kNN is gevoelig hiervoor. Als je heel veel negatieven hebt en maar een aantal plusjes, dan is het bijna onmogelijk om een correcte classificatie te gaan doen. Je bent niet gegarandeerd dat je een label kan gaan toekennen aan de klassevariabele.

**Slide 23:** Decision trees: heel krachtig en heel interpreteerbaar door mensen → worden heel vaak gebruikt.

C goed kennen want gegarandeerd oefeningen op het examen!

**Slide 24:** Stel je bent de voorbije weken weggegaan en je hebt ervaringen met drankjes opgedaan: soms was je ziek, soms niet. We gaan op een objectieve manier proberen in te schatten of het onbekende drankje tot een kater gaat leiden of niet.

**Slide 25:** We hebben een training set: we gaan kijken naar de karakteristieken van de drankjes, bv de kleur, vorm van het glas,...

**Slide 26:** Opdeling tussen glazen die (niet) rechthoekig zijn.

**Slide 27 :** We kunnen ook kleur gebruiken. Op die manier krijgen we een perfecte opdeling tussen drankjes die positief zijn en negatief.

**Slide 28:** We komen erbij uit dat het driehoekig is en een oranje kleur heeft, dus het leidt tot een kater.

**Slide 29:** Een decision tree heeft een aantal elementen:

* Internal node: staan ten opzichte van leaf node (de gezichtjes in dit geval).

Test op een attribuut.

* Leaf node: verdeling van instances volgens je klassenlabels.
* Branch: pijlen: waarde voor een bepaald attribuut.

In elke internal node kies je 1 attribuut om je instances op te splitsen. Wanneer je een nieuwe case gaat classificeren, ga je die 3 volgen en ga je een label toekennen aan de nieuwe instance.

**Slide 30:** We gaan attributen selecteren om de observaties in de dataset zo goed mogelijk op te splitsen. Zo goed mogelijk: op zo’n manier dat we de 2 soorten klassen zo goed mogelijk uit elkaar kunnen halen. We gaan ervoor zorgen dat de leaf nodes zo zuiver mogelijk zijn (→ zoveel mogelijk observaties bevatten van 1 klasse). Dat attribuut selecteren dat ervoor zorgt dat het resultaat van die splitsing voor zuiverdere leaf nodes zorgt. We gaan onzuiverheden berekenen: impurity. 3 belangrijke metrieken, entropy en gini zijn het meest gebruikt. We gaan die niet bekijken, maar je moet wel weten dat ze bestaan. Wij gaan werken met de classification error. Het is de bedoeling een inschatting te maken hoe zuiver een leaf node is.

**Slide 31:** We gaan het omgekeerde doen van wat er op de slide staat: we vertrekken van een lage purity en we willen op een hoger niveau geraken, veel zuiverder (dus woorden moeten eigenlijk omgekeerd staan, of afbeelding ondersteboven zetten). We gaan waarden berekenen en kijken naar waarden om in te kunnen schatten of een leaf node zuiver is of niet.

**Slide 32:** Stel dat je in een leaf node terecht komt die de gegeven verdeling heeft (eerste kader). Je gaat het maximum nemen van de twee verhoudingen die er zijn. Hoe zuiverder de node, hoe lager de impurity score. Ergst mogelijke is de laatste verdeling: 50/50: heel onzuivere split. Je krijgt dan maximale waarden, bij error en gini is dat 0.5.

**Slide 33:** Uitgelegd in termen van fractie van bepaalde klasse. Het gaat hier om een binair probleem. Hoe meer je in het midden zit, hoe hoger de impurity.

**Slide 3§:** We willen de impurity berekenen van een split. We kunnen meerdere attributen gebruiken om te splitsen en we willen dat attribuut gebruiken dat het beste is. De variabele die leidt tot de laagste impurity is de beste split.

**Slide 3ç:** Purity berekenen van de split: 1-max(fracties). Zie Figuur 21.2.

We doen hetzelfde voor B en vinden dat B de laagste impurity heeft. We gaan enkel splitsen wanneer we de purity daarmee kunnen verbeteren. We moeten de impurity kennen van de root node → impurity van voor de split kennen.

**Slide 41:** Als de impurity hetzelde blijft, is het niet nuttig van te splitsen.



Figure 21.2: Berekening impurity voor A.

**Chapter 22**

# Les 19: 08/05/2015

## 22.1 Slides: 6B.BusinessIntelligenceandDataAnalyticsstudentversion

**VANAF HIER IS ALLES VAAG MET DE SLIDES**

**Slide 44:** Voorbeeld van hoe we zo’n beslissingsboom gaan bouwen op basis van een dataset. We moeten de impurity berekenen van de leaf nodes om de impurity van de split te berekenen. Kan ook op het examen terugkomen! Kan gevraagd worden wat het beste attribuut is om te starten als je een decision tree wilt bouwen.

We hebben 4 mogelijke waarden en 1 predictive variable. Hoe begin je daaraan?

1. 4 mogelijke variabelen: exhaustief testen wat de impurity verbetering zou zijn als we daarop zouden splitsen. Je neemt 1 van die attributen, bv Outlook, checken hoeveel mogelijke variabelen die heeft. Zie Figuur 22.1.

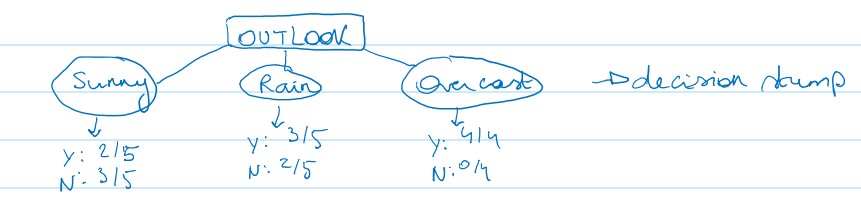


Figure 22.1: Kijk hoe de verdeling van elk van de variabelen van Outlook is.

1. Berekenen wat de impurity score is op basis van Outlook. Aan de hand van classification error metriek: zie Figuur 22.2.
2. Neem impurity van de split: zie Figuur 22.3.

Volg deze 3 stappen voor elk van de variabelen. Degene met de laagste impurity is het beste attribuut om op te splitsen.

**Slide 45:** Humidity en windy zullen leiden tot hogere impurity dan outlook, dus splitsen we daarop.

**Slide :** Decision tree vervolledigen.

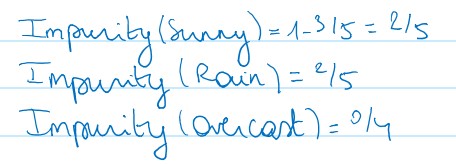


Figure 22.2: Bereken de impurity scores op basis van Outlook.

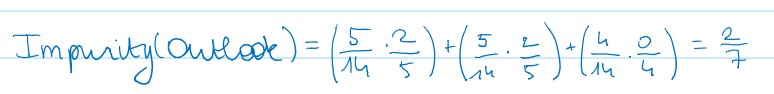


Figure 22.3: Bereken de impurity van de split.

**Slide 46 ev:** Zie Figuur 22.4, 22.5, 22.6, 22.7, 22.8, 22.9 en 22.10.

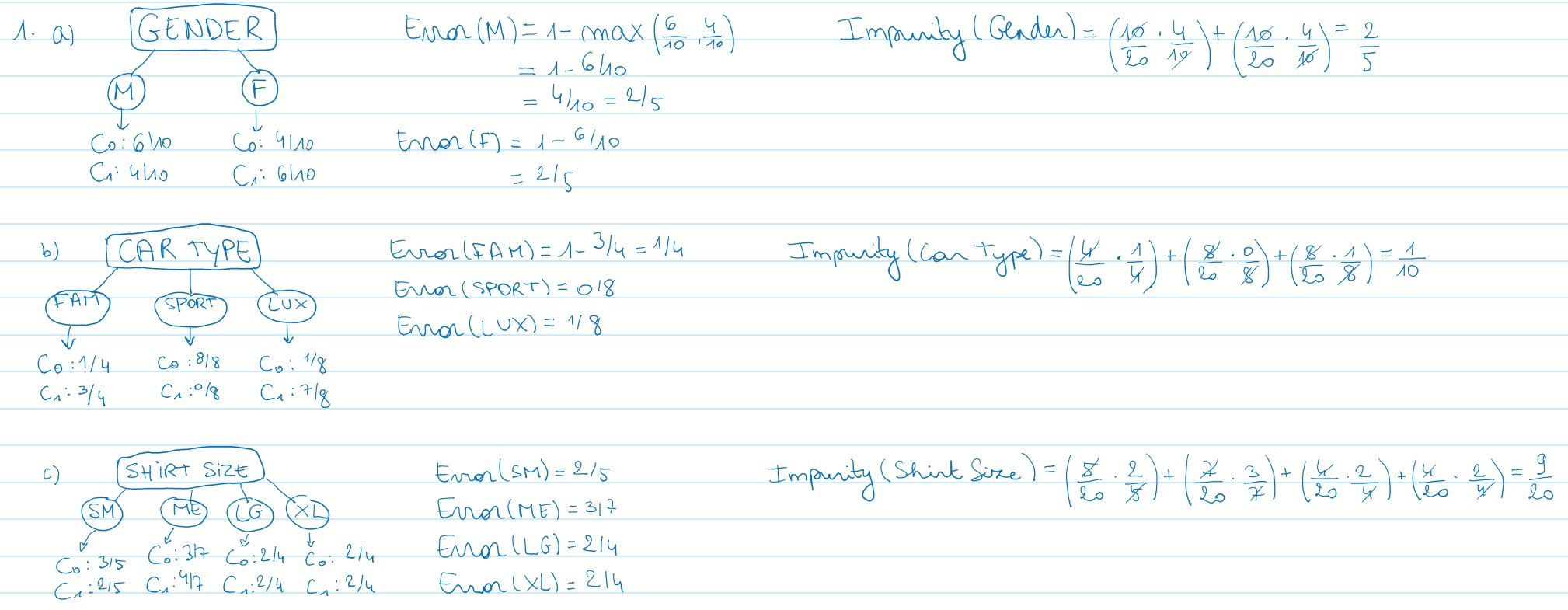


Figure 22.4: Suppose that you want to build a decision tree to predict the class label C0 or C1. Compute the impurity of the first split for attributes (based on classification error)

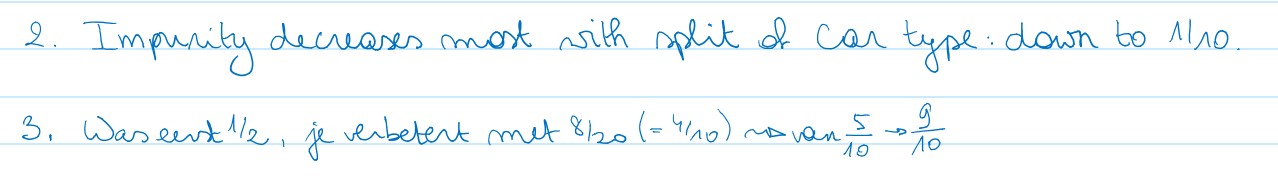


Figure 22.5: 2. Which of the three attributes would be the best choice as splitting attribute for the first split in the tree? & 3. What is the purity increase realized by splitting the instances based on this best splitting attribute?

**Slide :** Extra elementen: classificeren van nieuwe instances: toepassen van decision tree op labels die je nog niet kent. Je kan zo ook voorspellen of iemand gaat churnen. Voorspellen wat je gaat antwoorden op een ”nieuwe” vraag. Is basisprocedure voor de volgende slide.

**Slide :** Inschatten kwaliteit decision tree. We gaan het model evalueren. We gaan instances classificeren waarvan we het label kennen. We noemen zo’n

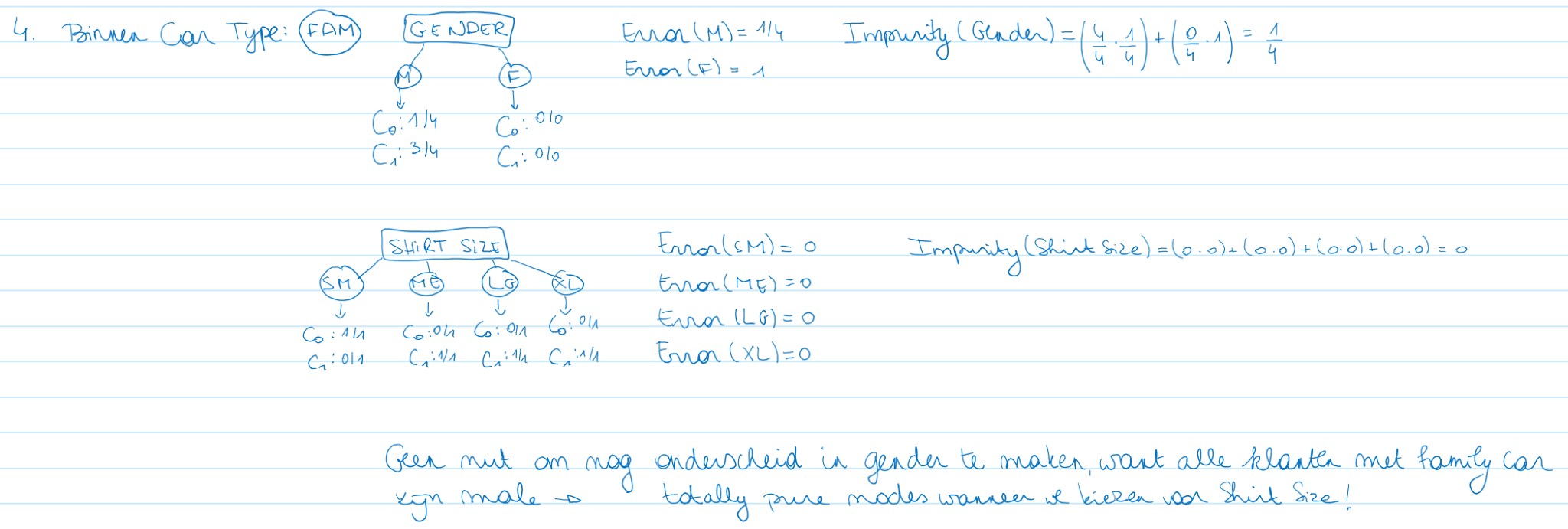


Figure 22.6: Complete the decision tree (1).

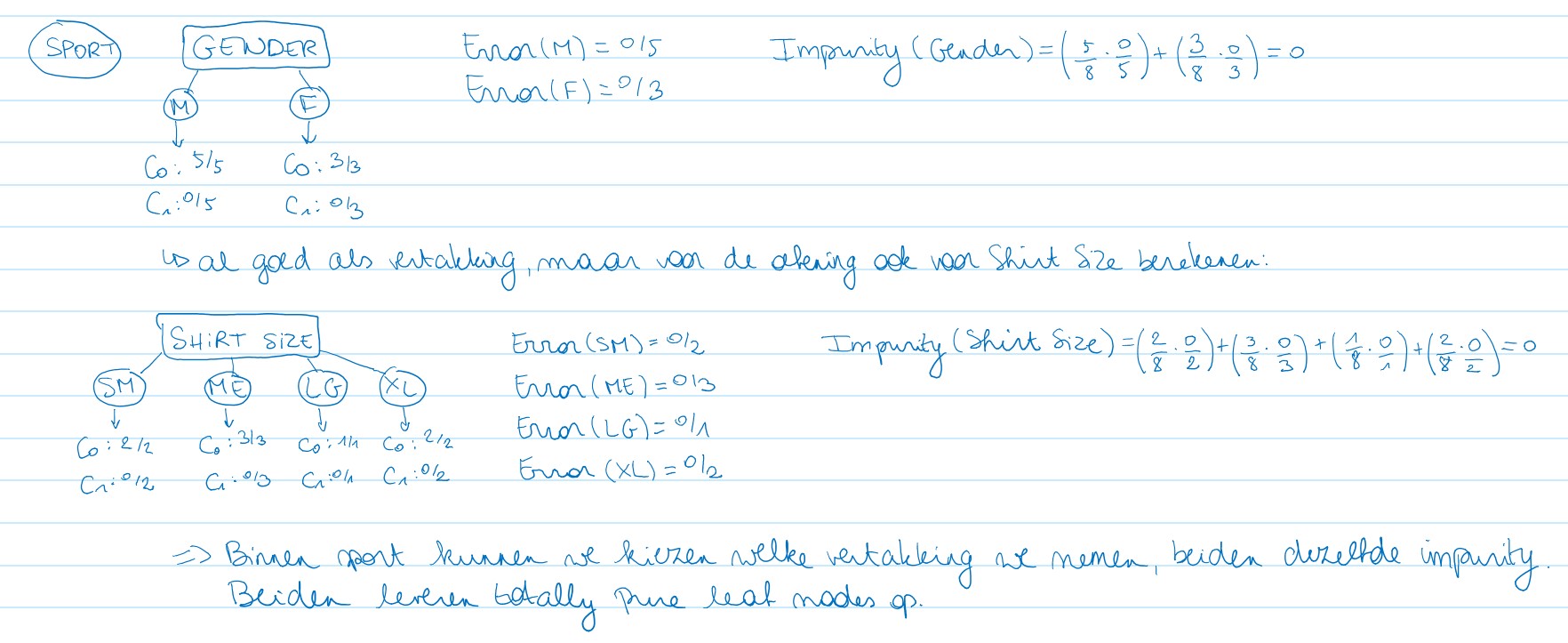


Figure 22.7: Complete the decision tree (2).

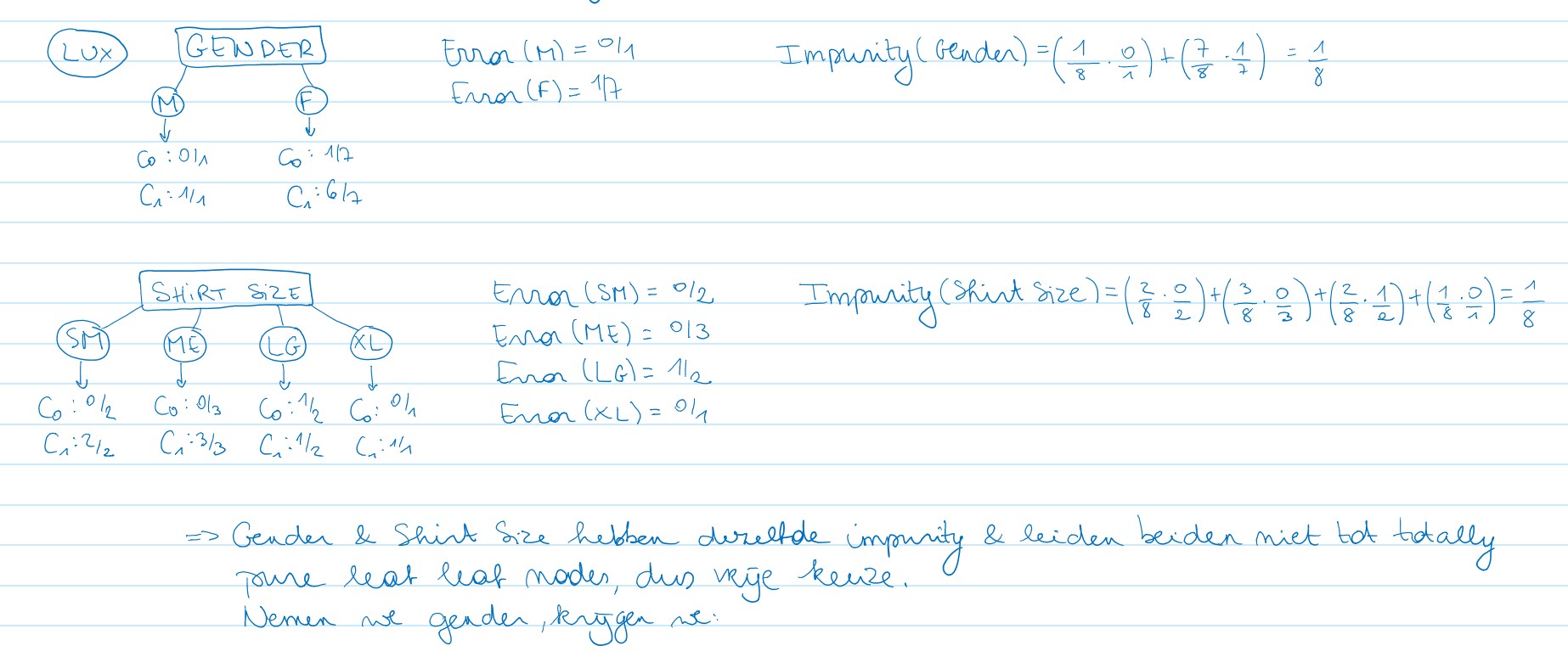


Figure 22.8: Complete the decision tree (3).

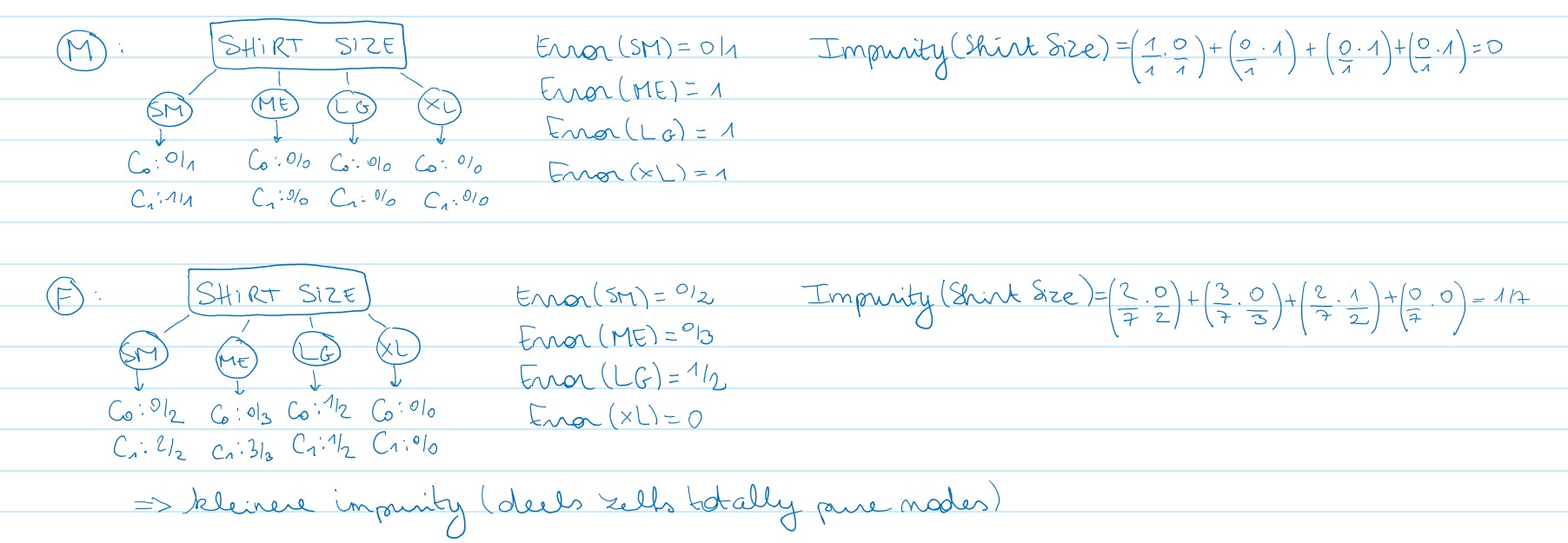


Figure 22.9: Complete the decision tree (4).

set van observaties een test set.

Training data: om het model te leren, om de decision tree op te stellen. Typisch gaan we 1/3 van onze oorspronkelijke data aan de kant houden om ons algoritme te evalueren: unseen examples. Het algoritme heeft die data nog niet gezien.

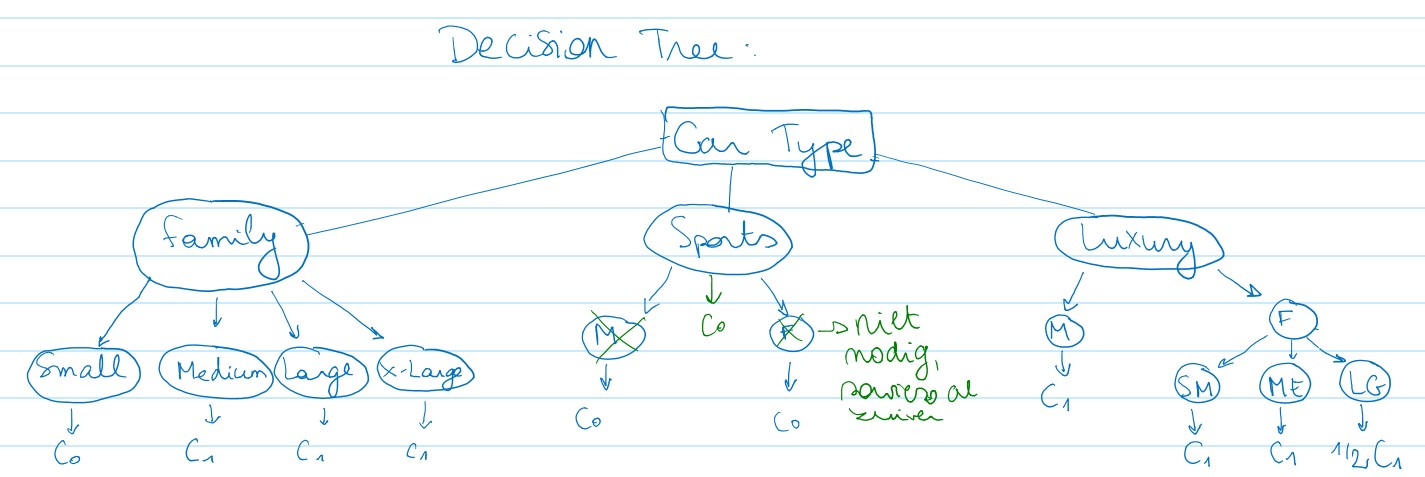


Figure 22.10: Complete the decision tree (5).

Hoe? We kennen de waarden voor onze predictieve variabelen. De oefening bestaat erin om de predicties te vergelijken met het echte label. We kunnen deze observaties door het model jagen en daaruit krijgen we predicted class labels. Rechts in de onderste tabel staan de echte waarden. Geeft een indicatie hoe goed jouw tree is. Je wil dat al jouw predicted waarden dezelfde zijn als al jouw true class waarden → accuraatheid van 100%. De studie/het analyseren van deze 2 sets van waarden kan het best gedaan worden aan de hand van een schema, nl confusion matrix, zie volgende slide.

**Slide 50:** Confusion matrix doet vergelijking van predicted and actual class labels. Gaat tellen hoeveel keer er (in)correcte matches zijn. Wanneer we 2x yes hebben, noemen we dat true positives. Het tegenovergestelde daarvan zijn true negatives: twee keer no: het was negatief en we hebben negatief voorspeld. We kunnen ook misclassificaties hebben, noemen we false positives and false negatives. False negative: predicted class label is false, is no, terwijl het echte label yes is.

Als je die termen interpreteert, verwijst dat steeds naar de predicted class. Als we praten over een false negative, hebben we negatief voorspeld en het is foutief. Bij false positive hebben we positief voorspeld, maar is false (dus fout voorspeld). Als we dit berekend hebben, kunnen we de accuracy berekend hebben: tellen hoeveel keer we juist waren, gedeeld door alle observaties. Ook: error rate: hoeveel fouten hebben we gemaakt?

**Slide 5 ev:** Testen, zie Figuur 22.11

**Slide 59:** Tot nu toe: heel rooskleurig plaatje over hoe je een decision tree bouwt. Zolang je onzuivere leaf nodes hebt: verderbouwen. Je gaat altijd zo ver mogelijk bouwen. → Is een probleem en als je zo’n algoritme zou specifi¨eren, zou je geen goede classificatiemodellen bekomen. Wordt in de figuur op de slide getoond: toont de accuracy op de verticale as tegenover de grootte van de tree. Hoe groter je je tree gaat bouwen, hoe meer rechts je terecht komt. 2 waarden geplot: accuracy op de training data en accuracy op de test data. Uit de training data kan je afleiden dat we in staat zijn om de accuraatheid op de trainingdata altijd te verbeteren. Als we de accuraatheid zouden berekenen op basis van die instances om de tree te bouwen, kunnen we die alleen maar verbeteren. Het probleem zit ’m in de test data: naarmate de tree groter wordt, gaat de test data accuracy dalen. Is een probleem want accuraatheid van de tree wordt gebaseerd op de test data. Komt door het probleem van overfitting: je decision tree wordt

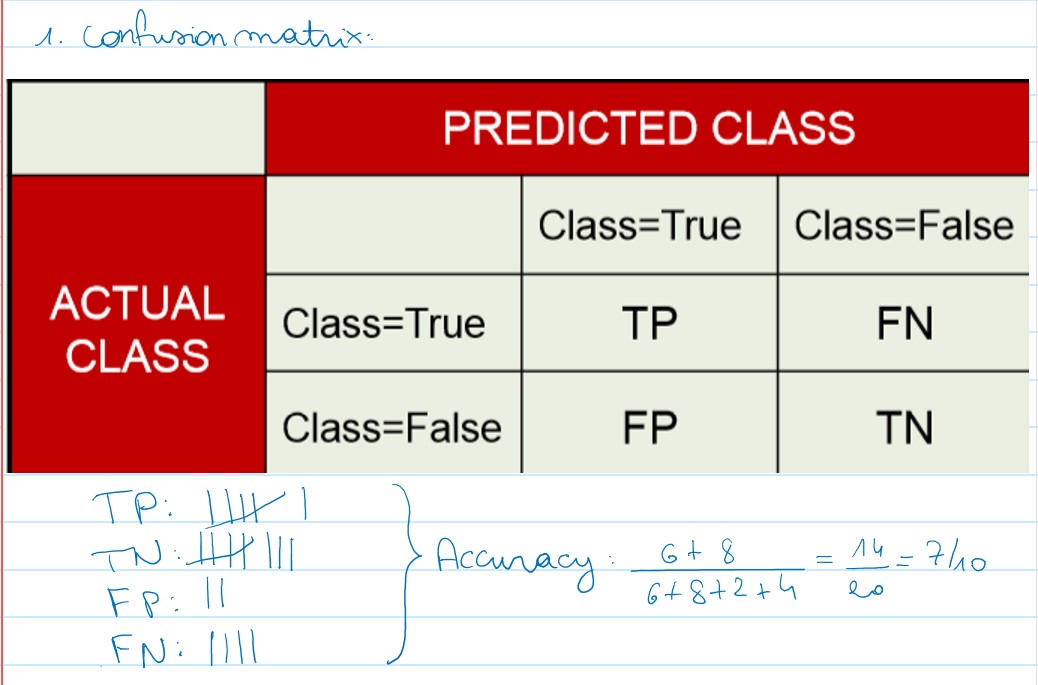


Figure 22.11: Confusion matrix.

te specifiek voor de training data. Het gaat karakteristieken modelleren die eigen zijn aan de training data en die niet generiek zijn en niet toepasbaar zijn op data die je nog niet gezien hebt. Te specifiek: wordt aangegeven door de gap.

Het tegenovergestelde verhaal is underfitting: je leert niet ver/lang genoeg, je maakt je boom niet groot genoeg. Is de linkerkant (grijze verticale streep) van het verhaal. Op die manier bekom je zowel een lage training data accuracy als een lage test data accuracy.

Wat is nu het optimale model? Als we puur naar accuracy gaan kijken, zijn de beste trees degenen waarvoor de test accuracy maximaal is, in de groene ovaal op de tekening. Het evalueren van welke tree we zouden willen hangt niet noodzakelijk enkel af van accuracy. In sommige contexten ga je toch een grotere, minder accurate tree nemen, maar niet vaak. Het omgekeerde wel: kleinere boom die niet het meest performant is in termen van accuraatheid, maar kan nodig zijn voor duidelijkheid/leesbaarheid. Accuraatheid is dus niet het enige principe. Op basis van accuraatheid zou je in dit vb dus kiezen voor een tree die tussen de 10 en 20 ligt.

**Slide 60:** Op een bepaald niveau moet je abstractie kunnen maken, kunnen generaliseren. Als je gaat overfitten, ga je noise modelleren. Op de linkergrafiek staat een noise point die eigenlijk foutief is en die heel onwaarschijnlijk is om correct te zijn. Als je classificatiemodel toch zou rekening houden met dit noise point en zou modelleren zoals de dikke groene lijnen, ga je spreken van een overfitting model. Het zou eigenlijk het lineair model moeten zijn. Rechtse tekening: het zou de zwarte lijn moeten zijn, de groene lijn is te overfitting, gaat modelleren dat je eigenlijk jouw decision boundary veel te specifiek/nauwkeurig voor de training data gaat maken. Stel dat je binnen de groene lijn op het rode puntje aan de blauwe kant een nieuwe observatie krijgt. Het is het meest waarschijnlijk dat je daar blauw krijgt. Als je dat groen model zou volgen, zou het als rood voorspeld worden. Daar wordt jouw model te specifiek voor jouw data en niet specifiek genoeg voor de data die je nog moet zien.

**Slide 61:** Hoe doen we dat nu met decision trees? Hoe verhinderen dat die bomen veel te groot worden?

Je hebt 2 soorten methoden om decision trees niet te laten overfitten:

1. Stop met bouwen: op basis van een of ander criterium ga je beslissen om te stoppen. Bv kijken naar de split die je krijgt, is de impurity verbetering nog goed genoeg, is die verbetering niet te klein? Als deze kleiner is dan 10%, hou dan de boom bv kleiner. Je kan ook stoppen wanneer boom 10 splits krijgt: maximale grootte.
2. Pruning: pas het algoritme zoals eerder uitgelegd toe, maak de boom zo maximaal en zo zuiver mogelijk. Nadat je die gebouwd hebt, ga je takken wegsnijden. Je gaat op een bepaald moment een stuk van de tree (subtree) ”afknippen” en groeperen in een leaf node. Alle instances die in die subtree zaten, ga je samenbrengen en 1 leaf node overhouden.

**Slide 62 & 63:** Voorbeeldvraag van het examen van vorig jaar: D: Decision tree A is underfitting, C is overfitting. Hier zitten we met een error percentage, dus het omgekeerde van accuracy. We zitten hier met een error percentage dat vrij hoog is. → Vrij lage accuracy. Aangezien we nog kunnen verbeteren, spreken we bij A over underfitting. C is inderdaad overfitting: we zitten met een error rate die terug naar boven gaat. Is hetzelfde als daarnet: accuracy rate die naar beneden ging.

**Slide 66:** Belangrijke slide! Kunnen vergelijken met de voor- en nadelen van k-nearest neighbours. Voordelen decision trees:

⊕ Inexpensive to construct: het kost niet veel rekentijd om zo’n decision tree te bouwen, snelle en performante techniek, hoe snel je een model kan bouwen/maken.

Ook het geval voor k-nearest neighbours.

⊕ Extremely fast at classifying unknown records: extreem snel in het classificeren van nieuwe instances. Bij k-nearest neighbours: alle afstanden naar alle andere instances berekenen. Bij decision trees is het super eenvoudig. ⊕ Easy to interpret for small-sized trees: voor relatief kleine bomen (t.e.m. 10, maximaal 20 splits) zijn ze heel begrijpbaar voor mensen. Is een verschil met k-nearest neighbours en andere, meer complexe technieken (neurale netwerken,...): die zijn heel moeilijk interpreteerbaar. We noemen deze (moeilijke) modellen black box technieken. Decision tree is white box, want we kunnen heel duidelijk interpreteren wat erin staat.

⊕ Accuracy is comparable to other classification techniques for many simple data sets: zeer goed vergelijkbaar met andere, complexere technieken. Ook een belangrijke reden dat de techniek vaak toegepast wordt.

## 22.2 Slides: 6C.BusinessIntelligenceandDataAnalyticsstudent version

**Slide 2:** Bij predictieve data mining hebben we de voorspellende variabele (variable of interest), bij descriptieve niet. Hierbij gaan we op zoek naar patronen in de data zonder op voorhand te weten welke variabele een belangrijke rol gaat krijgen. Alle variabelen spelen dezelfde rol, hebben hetzelfde belang.

**Slide 3:** Bij predictive data: labeled data set. Bij descriptive: unlabeled. We gaan hierbij kijken naar 2 technieken: clustering en association rules learning.

**Slide 5:** Clustering: in essentie: het vinden van homogene groepen van instances in een data set. Die doelstelling leidt tot 2 targets:

* Je moet ervoor zorgen dat de afstanden tussen de elementen in 1 cluster zo minimaal mogelijk zijn.
* Afstanden tussen clusters moeten zo groot mogelijk zijn.

Wat wil dit zeggen in termen van een data set? Elke observatie in een data set heeft bepaalde waarden. Als je daar een afstandsfunctie over defini¨eert, over die waarden, dan kan je een afstand daarover defini¨eren en daar gaat het hier over. In termen van de attribuutwaarden: je wil bij voorkeur dat observaties die in 1 cluster horen, zoveel mogelijk gelijke waarden hebben. Tegelijk willen we ervoor zorgen dat de data waarden tussen verschillende clusters zo ver mogelijk uit elkaar liggen.

**Slide 6:** Probleem met clusters: niet altijd eenvoudig te weten hoeveel je er zoekt. Je moet de evaluatie gaan maken: wat is de beste structuur die we kunnen vinden?

**Slide 7:** Als we spreken over clustering, spreken we over een aantal clusters. We maken een verschil tussen partitionele en hi¨erarchische clustering.

**Slide 8:** Bij hi¨erarchische clustering ga je een hi¨erarchie van clusters krijgen. Bij partitionele clusters niet, je gaat een vlakke clusterstructuur krijgen, geen afhankelijkheden tussen clusters.

* Hi¨erarchische clustering:
  + Agglomeratieve hi¨erarchische clustering: je werkt bottom-up: vertrek van je instances en je gaat zoeken naar die 2 instances die het kortst bij elkaar liggen. Vervang dit dan door de gemiddelde waarde (centroid) en ga opnieuw kijken welke 2 clusters het kortst bij elkaar liggen. Je krijgt een hi¨erarchische structuur tussen clusters. Je cre¨eert een hi¨erarchische structuur. Hoe nu beslissen welke de uiteindelijke oplossing is? → Kijk hoeveel clusters je wil en op basis daarvan maak je een dwarsdoorsnede van het dendogram.
  + Devisive hierarchical clustering: top-down een hi¨erarchische structuur cre¨eren. Complexiteit die hierin bestaat is om die top-down clustering te specifi¨eren. Je gaat iteratief een partitioneel algoritme toepassen. Je start van al jouw instances en defini¨eert dan bv 2 clusters. Daarna splits je die nog verder op & op die manier krijg je een hi¨erarchische structuur → op die manier krijg je een devisive hierarchical clustering.
* Partitional clustering:
  + Distance based: gelijkaardig aan agglomerative hierarchical clustering, buiten dat er geen hi¨erarchie is. Je hebt weer je ruimte van instances en gaat clusters defini¨eren. In plaats van bottom-up te starten, gaat k-means anders werken: zegt eerst hoeveel clusters je wil. Je gaat dan op een random manier 3 seeds bepalen. Vervolgens ga je voor elk van de andere observaties die observaties toekennen aan de seed die het kortstbij is. Op die manier krijg je een vlakke clusterstructuur. Zijn perfect onafhankelijk van elkaar.
  + Density based: hetzelfde als distance based, maar dan kijken naar de densiteit.

Op zich is elk element een cluster op zich!

**Slide 9:** Afbeelding is stock clustering: clusteren van aandelen. Evolutie van de prijs van het aandeel wordt ook vaak gedaan. Aandelen uit clusters combineren om een evenwichtige portefeuille te bekomen.

**Chapter 23**

# Lecture: Business Information Systems 9-1: E-business v Les 20

## 23.1 Slides: 7A.E business part1

### 23.1.1 Video: Business Information Systems 9-1: E-business

**Slide 3:** Basic definitions: e-commerce: two sides: both sides can be efficiently automated by state-of-the-art IT, which we’re going to look at in this chapter. E-business: more broad. You can see that e-business is a lot more general in term of definition than e-commerce.

**Slide 4:**

* In terms of nature of participants:
  + B2C: for example: Dell.
  + B2B: Wall-Mart x Gilette for example.
  + C2C: eBay
* In terms of participants’ connection:
  + Through a PC.
  + Mobile commerce: smartphone. Closely related to that: Locationbased commerce. Imagine you walking with your smartphone in de Diestsestraat. Depending on your location there, you will have different ads on your phone.
  + Interactive digital TV: not very common.

**Slide 5:**

* E-learning: currently very hot. Toledo is an example of this. MOOX: courses offered on the Internet where people can register and have access to lectures,... → area which is becoming very popular.
* E-government: tax-on-web,...

⇒ E-business is really hot these days.

**Slide 6:** Before the Internet became popular, people would buy as shown in the slide.

**Slide 7:** Say you want to book a trip to Barcelona, in an Internet-environment, you’ll do it as shown. There’s a whole lot of information being gathered: customer preferences, browsing,... All that information can be analyzed. There’s a tremendous amount of data being gathered.

**Slide 8:** Services can typically be delivered 100% electronically. We cannot deliver a laptop electronically though. There’s only a few digitized items. For those products that need to be physically delivered, finding a cost-effective way of delivering it becomes a key issue.

**Slide 9:** Big difference between traditional & e-logistics: traditionally: stuff being sent to destinations (retailers, distribution stores,...). In an Internetenvironment, this has changed: delivered to the customer directly. Reverse logistics: if you send a product to the home of a customer and it turns out to be the wrong product, you need to get it back. Those facilities also need to be established.

3PL suppliers: UPS,... → companies that provide services to do logistics in an e-business environment.

**Slide 10:** E-commerce being categorized according to various dimensions:

* Virtual product dimension: we can have a digital product like an MP3, a book,... or a physical product, one that you need in a physical world: laptop, piece of clothing,...
* Virtual process: we can have a physical process, that means that we need a physical process of delivering the goods physically, or we can have a digital process, that means that the process is only digitally available, such that the goods can be delivered electronically.
* Virtual agent/player: we can have a physical agent, one that is available in the physical world, but we can also have a digital agent, which is only available on the Internet. Think about Amazon, that’s a digital agent.

There are various types of combinations between those 3 different types of dimensions. Pure electronic commerce typically means that you have a digital product, a digital process and a digital agent so everything can be done on a computer, on the Internet.

**Slide 11:** Different types of companies are now popular in our new economy:

* Bricks-and-mortar: old-economy companies, these are traditional companies based in the physical world only, they are built out of bricks and mortar, as the name suggests.
* Pure-play: virtual organization is a company that’s engaged only in electronic commerce, so it does not have any presence in the physical world. It is only engaged in electronic commerce. Examples here are Amazon, Facebook,...
* Clicks-and-mortar: these are combinations of both. They conduct some e-commerce activities, yet their primary business is done in the physical world. Many supermarkets offer e-shopping facilities. They do have some e-business activities, but their primary activities are done in the physical world.

**Slide 12:** Potential advantages:

* We don’t need a physical shop front anymore, just a website on which we can directly sell.
* Reduced transaction costs, especially when we have electronic delivery.
* Ease of crossing geographical boundaries.
* Websites available 24/7: the website is always available and to the whole world.
* Ease of updating existing and distributing new information: if there’s new books you want to sell on Amazon, it’s really easy: you just include the book information into the Amazon database and it gets automatically published on the Amazon website and it’s very easy to distribute that new information to our end consumers.
* Providing additional value for customers: you can actually provide targeted advertising. Amazon does this: based on your previous purchases and interests, Amazon is going to give you targeted recommendations based on your purchases and interests.
* Internet: universal, easy-to-use set of technology and standards: empowers smaller companies: can easily invest in e-business activities

→ There’s a lot of advantages for B2C e-commerce, but it doesn’t always have to be successful.

**Slide 13:** Dot com bubble burst: happened in the early 2000’s. we had all those companies jumping on the e-business bandwagon and we called them dot com companies, but not all of them were successful, because in the early days of e-business, people were still very reluctant to use it. A typical dot com company relied on network economies to build up market share. Amazon, when it started, not many people used it. Many people did not trust Amazon. But as soon as they had a critical customer base and those customers started giving positive feedback about it, Amazon started to gain money. You need those network effects, you need to make sure that enough people start to know you such that they can spread (positive) word of mouth so that more people will start making use of your website and you will start making money. But typically, at the start, you need to make sure that people start to know you. That’s not easy and it did not always work. Amazon, in its early years, had lots of losses, cause they had to give a lot away for free.

**Slide 14:** Pets.com: wanted to sell pet supplies online, but the business model was completely wrong. This is just one example, there exist plenty of examples like that. There was a vast increase in stock prices, until some e-companies went bankrupt → stock prices fell.

**Slide 15:** It was not always that successful. In the below graph, you can see some estimated e-commerce sales. You can see that it’s steadily increasing.

**Slide 16:** B2C is just one part of the story, we also have B2B, which is a much larger share of the revenue. B2B is a lot bigger and a lot more important than B2C, as you can see from the numbers on the slide. Note that B2B is not necessarily using the Internet and the WWW-technology. In B2B, other technologies, like EDI (Electronic Data Interchange) are also very popular.

**Slide 17:** B2B and B2C differences:

* More complex process that may involve extensive negotiation over prices, product specifications,... E.g. Ford has a GPS-supplier. They have to negotiate with the GPS-supplier about the product specifications: dimensions, functionality,... → Will typically involve the exchange of CAD/CAM-designs. Vs buying a laptop at Dell: a lot less complex.
* Often longer-term, higher stickiness. Stickiness: supplier and buyer engage into a long-term relationship. In B2C, you don’t have a high stickiness.
* Need for systems integration buyer/seller; additional technology.
* Make sure you have an agile setting in a B2B setting in which we can leverage cost efficiencies.
* Collaborative commerce: you’re going to have multiple partners involved. We can set up pretty efficient collaborations there by means of adequate IT to make sure that the whole process is as efficient as possible.

**Slide 18:** Key areas: a company typically has 2 sides: the end/consumer side and the supplier side. Companies can work according to a build-to-order or a ship-to-order philosophy. Build-to-order: you wait until it has been configured, then you’re going to build it and ship it to the consumer. A ship-to-order philosophy is what Amazon does: you order it and then Amazon will ship it to your home address. Every company typically has a B2C-side, which will typically be automated using CRM. A company also has a B2B-side, which will be optimized with SCM (Supply Chain Management). A company that is available online, needs to make sure that all its business processes are both internally integrated, as well as externally. Internal integration looks at the ERP, EAI (Enterprise Application Integration). Everything needs to be fully integrated, because integration is going to give you efficiency. External integration means that the firm should talk to the outside world, to its consumers and to its suppliers. Ideally, it should talk to this outside world using the optimal ICT-technologies.

**Slide 19:** Ideas that you also see popping up a lot are the idea of a front/back office.

* Back office operations: activities that support fulfillment of sales, such as accounting and logistics.
* Front office operations: business processes, such as sales and advertising, that are visible to customers. Everything what the end-consumer sees, everything that is fully visible to the consumer.

→ Everything should be fully integrated. Some examples of integration are shown on the slide.

**Slide 20:** E-business hasn’t always been this successful (dot com bubble burst,...). There are also technological and non-technological limitations.

**Slide 21:** There’s still a lot of socio-economic difference between communities in their access to computers and the Internet. Still, you sometimes have people that lack ICT-skills and literacy (refuseniks). The global digital divide: in certain regions in the world, Internet access is still quite difficult.

**Slide 23:** The Internet is changing the economics of information by:

* Shrinking information asymmetry: it’s easier for customers to obtain and compare pricing and other info. Think, for example, about the example given at the beginning: you want to plan a trip to Barcelona: if you want to compare different flights, you can do it on the Internet. It’s really easy to compare different information provided by different online companies: you just go to their website and compare the information.
* For the company itself, the Internet extends both the richness and the reach of information provision:
  + Cheaper way of reaching a wide audience than traditional advertising. An online company can now customize its website to its audience. There’s lots of opportunities for mass personalization, that means that the website can be customized to the needs of the individual customers,... → E.g. Amazon.
  + Opportunities for mass personalization.

**Slide 24:** The Internet has also reduced transaction costs, so:

* To find buyers, we don’t have to do mass-mailing of expensive brochures or expensive TV and radio ads. We can just customize our website, we can do mass-personalization.
* Fully automatic collection of payments: we can have online payment systems that allow us to collect payments automatically in a very efficient way.
* Delivering product, especially when electronic delivery is possible.
* Support.

**Slide 25:** Traditional economic cost typically looks like on the slide. You have the marginal revenue (the price), which is constant, we have the average total cost and the marginal cost, which keeps on going up. Now from traditional economics, we know the optimal number of units to be produced or sold is where the marginal cost equals the marginal revenue. This is a traditional economic cost structure.

**Slide 26:** The Internet is going to change this in terms of marginal economic costs: they will be almost constant, whether you’re selling 100 or 1000 MP3files, it’s going to cost you about the same. The marginal cost will be flat. You have nearly unlimited economies of scale so the Internet is definitely changing the way those costs are going to look like.

**Slide 27:** Cost structure of digital products or services:

* We have nearly unlimited economies of scale because we have no or few capacity constraints. In the old days, before the Internet, we had to invest in (manufacturing) buildings,... In an Internet environment, the only thing we have to do is invest in hardware services, which doesn’t cost a lot of money, compared to building an extra building.
* High fixed costs to set up the whole Internet/e-business infrastructure and • those fixed costs are incurred early, but may not be recoverable, but
* the marginal costs are typically approaching zero; remember the example of the MP3 files.

Given this cost structure, firms need to react to that and come up with new ways of competing on the Internet. Some ways of competing on the Internet are versioning, confounding and creating network effects:

**Versioning:** Companies are going to come up with different versions of their online services, whatever that may be. If you’re buying a piece of software online, companies are going to come out with new versions of that regularly so as to gain money. → One way of creating competitive advantage in an ebusiness environment, essentially by creating new versions of the software.

**Confounding:** Make it difficult for buyers to compare. Companies know that it’s very easy for their customers to compare on the Internet, so what they’re going to do is making it less straightforward to compare by offering an option that the competitor is not offering, so that it’s really hard to compare the offers of both companies, because they are never identically the same.

**Creating network effects:** Trying to establish that positive word-to-mouth, creating network effects, for example by investing in social ads (like on Facebook). It can start targeting its products or services to the friends of their customers using their Facebook information.

**Preemptive pricing:** Not discussed here.

Because the ways of doing business have changed, companies have to think of new ways of creating competitive advantage.

**Slide 28:** How is the Internet going to impact the value chain? Since the Internet offers a new distribution channel, it has a huge impact. The Internet has led to the unbundling of information from the traditional value chain channels. The terms discussed below this are shown in an example on the next slide.

**Slide 30:** Let’s take a company like Levi Strauss. In the old days of Levi Strauss (before the Internet), you had Levi Strauss as a manufacturer, they sent to clothes to the distributor, who then sent the clothes to the retailer and they sold it to the end consumers. Those retailers could be stores like Inno. Because of the emergence of the Internet, all of a sudden, Levi Strauss got direct access to its customers. They could set up a website: have direct access to its end consumers and directly start selling products to their customers. Distributors and retailers were no longer part of the game! This is what we call disintermediation: removal of intermediary steps in a value chain. That resulted in lower purchase and transaction costs because the distributor and the retailer were no longer part of the game and of course that could be reflected in price advantages. Being a distributor or retailer, you won’t like this. They started filing lawsuits against Levi Strauss. → Channel conflict. Many of these suppliers had to change their business models and had to stop selling their goods electronically and directly to their end consumers because the distributors and retailers weren’t happy with that. What many of those online retailers/manufacturers did was, instead of selling directly to the consumers, they came up with a website which allowed the customers to configure their product or services and once that was configured, they gave a link to the closest retailer where the customer could go and buy the product. → No longer selling products online, just links to nearest retailer store.

**Slide 31:** A channel conflict is tension among different distribution chains for the same product or service. That is something that can occur when a channel member perceives another channel to be engaged in behavior that prevents or impedes it from achieving its own goals. The web gives you a direct sales channel, so there we have the risk of alienating traditional sales reps or distributors. Lockout: threaten by not selling the product/services anymore.

Disintermediation: usually not instantaneous: typically takes place step by step, one is going to start by selling a few products/services electronically and is going to increase that step by step. The idea here is how to make sure that we can placate the partners in the distribution channel while taking steps toward the eventual demise of these relationships? That depends upon the balance of power: how much do we rely on our distributors or retailers to distribute our product?

The ultimate idea is channel cooperation: try to cooperate with your partners downstream the channel, try to find ways of collaborating with your manufacturers and retailers.

**Slide 32:** A famous example here is the case of Benjamin Moore Paints. It was a large paint producer in the US and worked together with a lot of doit-yourself businesses which basically refrained from the early adoption of ecommerce, because of the reasons listed. Because of the emergence of e-business, the website of Benjamin Moore was getting very popular and they could get direct access to their end consumers. They started direct selling through the Internet in 2010. Retailers of Benjamin Moore were not informed of this, so the CEO was required to rectify the move with a YouTube video. In that movie, the CEO states that they will no longer engage in directly selling paint to end consumers, but help consumers find paint they want and then give links to the nearest stores where they can purchase these things.

**Slide 33:** Reintermediation: there’s a new focus and there’s a new type of intermediary function in a value chain. That means that delivery becomes a critical part of the overall customer satisfaction and as we said earlier, with the emergence of e-business, we typically have small parcels that are sent to many homes. So we have reintermediation: new types of channel members that are going to get very important, new types of intermediaries: information brokers, e-marketplaces, delivery services, intelligent agents (shopping bud that is going to shop on your behalf or an aggregator that is going to aggregate different product or service requests in order to get a better deal).

**Slide 34:** How is the Internet and e-business going to change the way value is created within a particular industry branch? The way economical value is created within a particular industry branch is as follows: depends on internal rivalry between the existing competitors, the threat of new entrants, the bargaining power of customers, the threat of substitute products or services and to what extend suppliers can exert bargaining power. Those are 5 major forces that are going to determine how value is created or distributed within a particular industry branch.

**Slide 35:** How is the Internet going to have an impact on each of those five forces?

* Threat of substitute products or services: the proliferation of the Internet is going to create new substitution threats, it’s really very easy to come up with new products or services on the Internet.
* Buyers: it’s really easy to switch: you’re on the Internet, buying some product on Amazon and if for some reason you’re not happy with the products or services anymore, you can very easily switch from one side to another.
* Rivalry among existing competitors: the Internet allows you to widen your geographic market, to increase the number of competitors.
* Bargaining power of suppliers: think about the channel tension that we just mentioned: a seller now has direct access to its end consumers and gives it some extra power to the suppliers.
* Barriers to entry: the Internet is low barrier to entry: it’s very easy to enter the market: you just set up a website and you’re there.

**Slide 36:** The Internet has changed relations with suppliers and other business partners.

**Slide 37:** A business model is a way of gaining money, the method of doing business by which a company can sustain itself, creating added value from an economic perspective. It’s not always that straightforward to say who’s going to make money and how much should be charged for particular products or services.

**Slide 38:** Examples of common revenue models.

**Slide 39:** Thanks to the Internet, we have all kinds of new business models, like iTunes and Google,... You can also try to reinvent tried-and-true models (doing auctions, but in a different way than usually: eBay).

**Slide 40:** Brokerage: market makers: pieces of software that facilitate a particular transaction to take place.

**Slide 41:** Brokers are market makers: they bring buyers and sellers together and facilitate transactions and they can facilitate any aspect of the transaction. Usually a broker is going to charge a fee/commission for each transaction it enables.

**Slide 42:** Auction-brokers (like eBay). You have 2 different types of auctions: forward and reverse:

**Forward auction:** Typically what eBay is going to do. If I have a bottle of wine and I want to sell it to someone who likes wine, I can put it on eBay and people can make sequential bids. The one that is bidding the highest price is going to get the bottle. In a forward auction, there is an upward pressure on the price: you have one item and many interested parties can place bids.

**Reverse auction:** Ask different companies for a quote. Ask different companies how much they would offer for a product. In this setting, there’s a downward pressure on the price.

All of that can be facilitated by the Internet.

Liquidation auction: liquidate your stock, the stuff you want to get rid of (e.g. Dell wants to get rid of older models of its PC’s).

**Slide 43:**

* Proxy bidding: eBay does this. It’s not necessarily the money you’re going to end up paying, it’s just how much you would pay for it at most.
* Sniping: you don’t want to bid too early because then you’re pushing the price upwards, so you want to bid at the last moment.

**Slide 44:** There’s lots of opportunities for fraud and unfair practices. Puffing or shilling can be done in a forward and reverse auction.

* Non-acceptance of best bid: if you’re not happy with the highest price that was offered, retract the offer.
* Auction rings/bid shielding: fraudulent agreements between different people in order to get a fair deal. If I have my bottle of wine and there are 2 bidders who set up a fraudulent agreement: one of them bids 10 Euros for it and the other bids 1000 Euros. Those 2 are friends of one another and have an agreement: the one that is bidding a lot is just doing that to scare off other bidders. You only have 2 bids then. A couple of seconds before the auction ends, the guy offering 1000 Euros retracts its bid so there’s only the 10 Euro bid remaining. → You’re shielding your bid.

**Slide 45:** E-marketplaces that bring together buyers and sellers and offer all kinds of extra services on top of that.

**Slide 46:** Transaction broker: market maker that facilitates a particular type of transaction.

**Slide 47:** Facilities to do mobile/digital payments.

**Slide 49:** Direct model: manufacturer reaches buyers directly and thereby compresses the distribution channel, like Dell.

**Slide 52:** Crowdsourcing: relies on the wisdom of crowds. You’re going to try and achieve a particular aim by asking many people to contribute a small amount. That can be crowdsourced funding (Kickstarter). Crowdsourced work: outsource repetitive tasks.

**Chapter 24**

# Les 20: 11/05/2015

## 24.1 Slides: 6C.BusinessIntelligenceandDataAnalyticsstudent version

**Slide 10:** Association rule learning: tweede descriptieve data mining. Bij clustering moet je enkel het principe kennen, geen oefeningen kunnen. Bij association rule learning moet je wel oefeningen kunnen maken.

**Slide 11:** We zijn nog altijd bezig met het zoeken naar patronen in data sets en we weten dus niet waar we naar op zoek zijn: geen variable of interest. Geen specifieke variabele. We zoeken naar patronen/hidden structure in data. Het typische voorbeeld noemen we market basket analysis: analyseren van de inhouden van winkelkarretjes. Je gaat zoeken naar patronen: items die vaak samen gekocht worden: co-occurrence. We zoeken naar patronen die gelden voor een hele set van winkelkarretjes. Walmart merkte zo bv dat pampers en bier vaak samen gekocht werden. Ze zijn dat gaan uitbuiten en hebben de opstelling van hun rekken zo aangepast.

**Slide 12:** We gaan detecteren welke producten vaak samen gekocht worden. Dit is 1 applicatie van association rules. Je moet beseffen dat die association rule learning veel breder toepasbaar is.

**Slide 13:** We gaan in een dataset detecteren of bepaalde patronen zich voordoen. We gaan zoeken naar bepaalde patronen in een observatie.

**Slide 14:** We gaan zo’n dataset beschrijven als een transactiedatabase. Het lijkt sterk te verschillen van alle datasets die we tot nu toe hebben gehad: nu hebben we alleen items. Het is ook perfect te vertalen zoals een data set zoals we die vroeger hebben gezien. We hebben nog steeds een ID, maar we hebben hetzelfde patroon: we kunnen elk van die items als een attribuut gaan beschouwen, zie Tabel 24.1.

Het is niet zoveel verschillend, al lijkt het op basis van de slide wel verschillend. Wat we nu gaan doen is in zo’n dataset patronen zoeken naar items die vaak samen voorkomen. We zoeken naar een association rule.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Wine | Bread | ... |
| 1 | 1 | 0 | 1 |
| 2 | 0 | 0 | 1 |

Table 24.1: Transactiedataset

**Slide 13:** Implicatie expressie in de vorm van *X* → *Y* : X leidt tot Y. Voorbeeld: bread en butter leiden tot milk. Als bread en butter voorkomen in een transactie, dan ook milk. Zo moet je zo’n associatieregel lezen.

Het is de structuur van patronen die we willen gaan ontdekken in zo’n dataset. Hoe gaan we dat doen? 2 basismetrieken die aangeven of een associatieregel interessant is. Er zijn eigenlijk maar een beperkt aantal associatieregels die interessant zijn. De kwaliteit van zo’n regel wordt gekenmerkt door 2 metrieken: support en confidence: daar ga je mee zien of een patroon interessant is. Stel dat je 2 verzamelingen hebt: A en B en je hebt elementen in beiden. Sommigen komen overeen. *A* ∪ *B* is gelijk aan de verzamelingen van alle elementen van A en B. Functie sigma: kijken hoeveel keer een element voorkomt in een transactie, waarbij die transactie *ti* een element is van alle transacties. In het voorbeeld gaan we kijken hoeveel keer bread, butter en milk samen voorkomen in elk van de transacties in **Slide 14**.

Voor support tel je dan voor hoeveel transacties deze elementen (allemaal) een deelverzameling zijn van de transacties (per transactie). Dat deel je dan door het aantal transacties.

Confidence: je gaat tellen hoeveel keer X een deelverzameling is van de transacties. Je gaat kijken hoeveel keer bread en butter een deelverzameing is van de transacties.

**Slide 19 ev:** Oefening 1: zie Figuur 24.1.

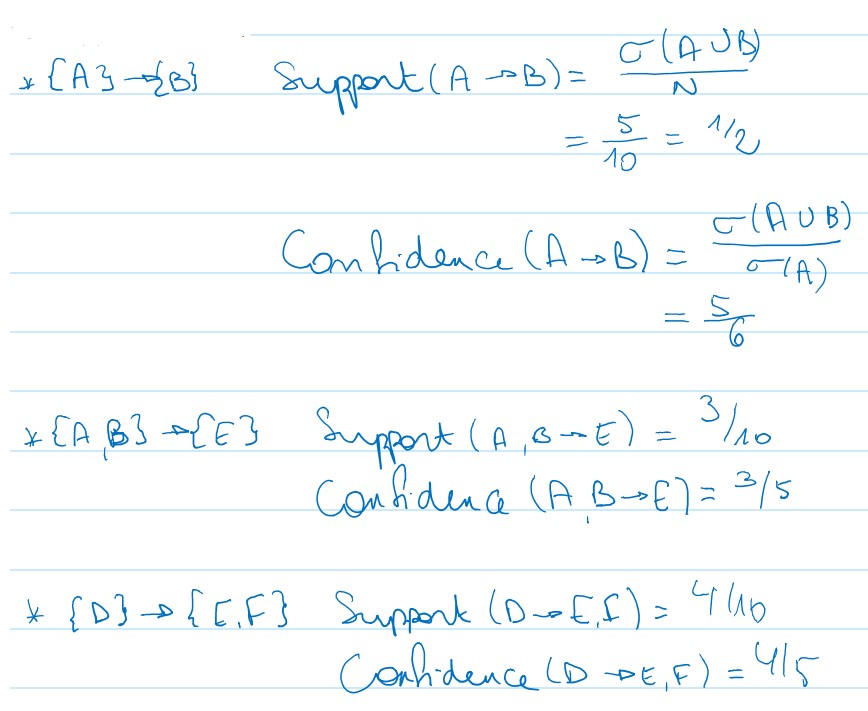


Figure 24.1: Oefening 1

**Slide 15:** We moeten association rules ontdekken in data, we gaan die niet zomaar krijgen. Je kan er bijna oneindig veel ontdekken, dus je moet een of ander algoritme hebben om die associatieregels te gaan ontdekken met de hoogst mogelijke support. Je moet een minimum support en minimum confidence opgeven.

Dan kan dat algoritme aan de start gaan. Er zijn 2 stappen:

1. Genereer alle item sets die voldoen aan de minimum support requirements. De item set moet een voldoende hoge support hebben. We gaan dus eerst item sets genereren/onderzoeken die voldoen aan de minimum support. Als je die vindt, kan je pas stap 2 uitvoeren.
2. Genereer high confidence rules.

**Slide 21:** Oefening 2: zie Figuur **??**.

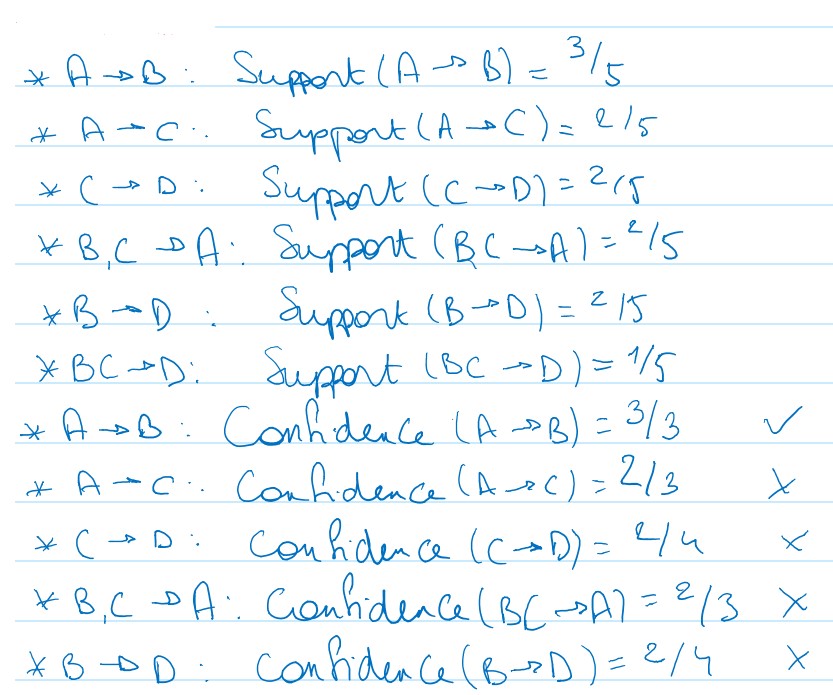


Figure 24.2: Oefening 2

Ook slim om van elke verzameling met 1 element de support te berekenen. Op die manier kan je zien waarvoor het nuttig is om te gaan combineren, zie Figuur 24.3.

Hieruit weten we dat E niet bruikbaar is, want de support daarvoor is al te laag.

Nota bij mijn oplossing: alle combinaties afgaan waarvoor de support hoog genoeg is.

**Slide 16:** Oorzakelijk verband: een association rule does not necessarily imply causality! Het is gevaarlijk van dat causaal verband meteen af te leiden. Het is niet noodzakelijk zo dat een associatieregel doelt op een oorzakelijk verband. Associatieregels kunnen enkel maar zekerheid geven over het samen voorkomen, maar niet op causaliteit.

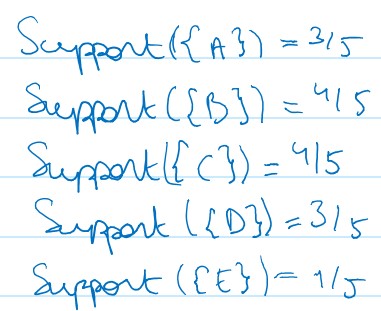


Figure 24.3: Oefening 2, support per item.

**Slide 17:** De confidence is in sommige gevallen een slechte indicator om te kunnen inschatten of een associatieregel goed is of slecht. Stel dat we de regel thee leidt tot koffie gaan bekijken, komen we een confidence van 75% uit. Je moet beseffen dat die confidence metriek misleidend is, want als we enkel naar koffie gaan kijken, zien we dat koffie in 9/10 winkelkarretjes voorkomt. Je confidence is 75%, maar zonder dat je weet dat iemand thee koopt, ben je 90% zeker dat iemand koffie zal kopen. → confidence is geen goede metriek. **Slide 18** toont een betere metriek.

**Slide 18:** Interestingness measure: maakt een afweging tussen substitutie en complementariteit. Thee en koffie zijn substituten van elkaar. De interestingness measure geeft aan wanneer dat zo is. Als de interestingness hoger is dan 1, weten we dat ze complementen zijn van elkaar. Bij het thee/koffiegeval gaat de interestingness 0.8333 zijn, dus *<* 1, dus zie je dat ze geen complementen zijn.

**Slide 26:** Examenvraag van vorig jaar.

Antwoord is C: Je moet op zoek gaan naar die item stets van grootte 3 waarvoor de support value lager is dan 50%. Ofwel weet je zelf al dat er 10 item sets zijn van grootte 3, als we deze 5 items (A,B,C,D,E) hebben. We kunnen ook gewoon alles oplijsten: alle mogelijke item sets van grootte 3 oplijsten. Als je die allemaal lijst, kan je per item set nagaan of dat leidt of je de support nog moet testen. A heeft een lagere support dan 50%, dus alle item sets met A gaan *<* 50% zijn. Je gaat enkel B,C,D nog moeten checken. Uitgewerkt: zie Figuur 24.4.

→ per item set kijken of support *<* 50%. ⇒ Alles met A al zeker (groen omcirkeld). Voor de rest verder kijken naar de tabel naar de mogelijkheden.

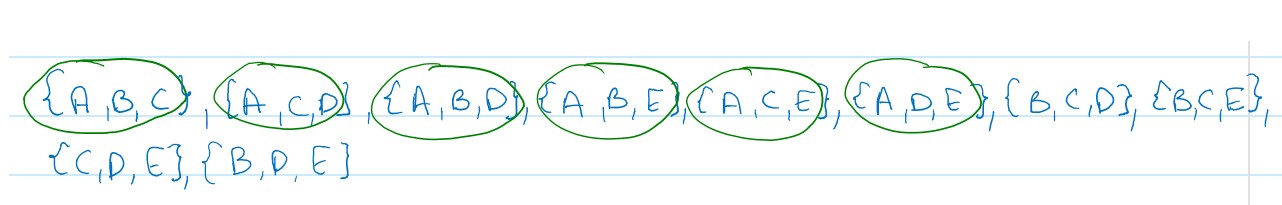


Figure 24.4: Uitgewerkt.

## 24.2 Slides: 7D.E-business-part1-QuizSession-Student Version

Keyword search and linking: we krijgen een keyword (bv Leuven) en we moeten een linking word zoeken. Op het moment dat je dat woord zegt, moet je in 1 zin uitleg geven waarom dit woord bij het keyword hoort.

**Leuven:**

* KULeuven: je kan hier studeren aan de universiteit.
* Bier: AB InBev fabriek.
* Tobback: burgemeester van Leuven.
* Belgium: stad in Belgi¨e.
* OHL: voetbalploeg van Belgi¨e.

**Dot Com Bubble Burst:**

* Stock Market: is gecrasht.
* Late 2000: toen deed zich de bubble burst voor.
* Mind share: een bedrijf wil marktaandeel verwerven. De bedrijven die daar failliet zijn gegaan, waren niet noodzakelijk gefocused op market share, maar op mind share: het was voldoende dat die producten in de hoofden van mensen waren om de waarde van dat bedrijf omhoog te drijven. Ze wilden een zo groot mogelijke klantenbasis verwerven. → reden waarom bubble ontploft is.
* Network economies: de bedrijven vertrouwden op netwerkeffecten (Facebook, Twitter,...) → zorgen dat ze een zo groot mogelijk netwerk cre¨eren om de concurrentie te snel af te zijn.
* B2C: het ging voornamelijk over B2C-bedrijven.

**B2B:**

* Business-to-Business: B2B is hier de afkorting van.
* Long term: langetermijnrelaties tussen bedrijven.
* Stickiness: het feit dat bedrijven zich aan de hand van technologie gaan integreren, hun systemen op elkaar gaan afstellen en dan een langetermijnrelatie gaan aangaan die heel strict is, waar bedrijven moeilijk onderuit kunnen.
* Systems Integration.
* EDI: Electronic Data Interchange: basistechnologie voor B2B-relaties.

**Channel conflict:**

* Disintermediation: tussenpersonen weghalen door web-based direct sales.
* Channel cooperation: distributeurs toch een kans geven om te verkopen: afstemmen op elkaar door bv afspraken te maken over de prijs of mee inschakelen bij de web-based sales.
* Value Chain.
* Web-based direct sales.
* Re-intermediation: nieuwe partners in de value chain introduceren (DHL, UPS,...). Soms omdat deze noodzakelijk zijn.

**External/internal conflicts uitleg:**

**Interne conflicten:** Binnen het bedrijf. Een bedrijf kan werken met verkoopsverantwoordelijken (deur aan deur bv), deze mensen zijn intern aan de organisatie. Als je direct gaat verkopen aan de klant via een website, ga je in conflict gaan met de mensen binnen je bedrijf.

**Extern:** Mensen die niet in jouw bedrijf actief zijn.

**Brokers:**

* Auction brokers: realiseren de verkoop van producten/maken dat mogelijk.
* Transaction brokers: vergemakkelijken transacties zoals betalingen. Zijn ook valide in een niet-Internet context.
* Market maker: vraag en aanbod naar producten samenbrengen of het mogelijk maken dat kopers en verkopers elkaar vinden.
* B2B E-marketplace: broker tussen bedrijven in om B2B mogelijk te maken.
* Commission: geld dat ze opstrijken voor hun diensten.

**E-auction:**

* Forward auctions: 1 verkoper en meerdere kopers.
* Reverse auctions: 1 koper en meerdere verkopers: de koper gaat een markt cre¨eren door te afficheren welke producten hij wil kopen en dan mogen de verkopers producten aanbieden.
* Proxy bidding: typisch gebruikt op eBay.
* Sniping: op het laatste moment bieden, zo lang mogelijk wachten → niet illegaal.
* Shilling/puffing: gebruik maken van partner in crime.

**Chapter 25**

# Lecture: Business Information Systems 9-2: E-business vLes 21

## 25.1 Slides: 7B. E business part2

### 25.1.1 Video: Business Information Systems 9-2: E-business

**Slide 1:** We’re going to continue discussing B2B technologies and business models.

**Slide 2:** In the previous section, we already clarified B2C and we briefly touched upon B2B. We also mentioned that B2B is totally different than B2C. B2B involves long-term relationships, more complex relationships and often also a higher stickiness. We also indicated that B2B usually has a much larger share of e-business than B2C. In what follows, we’re going to give an introduction to B2B and discuss the changing supply chain. Then we’re going to talk about B2B technologies: Traditional EDI and Internet/XML-based frameworks. That will be followed by B2B models. We’re going to zoom into sell-side B2B; buyside B2B and (public) e-marketplaces and we’re going to conclude with private industrial networks.

**Slide 3:** Here’s the setting that we’re going to use as a framework to discuss the remainder of this section. Companies can operate according to a build-toorder philosophy, like Dell, where products are being built as they are ordered by the customers. They can also operate according to a ship-to-order philosophy whereby products are being shipped as they are being ordered by the customers. Remember that every company has 2 sides: a B2C side, which is the interface to the customers, to consumers. We discussed this in the previous section. It also has a B2B side, which is a supply chain management side. That will be the focus of this particular section: the B2B side, which is the interaction with the suppliers.

**Slide 4:** Supply chain management definition: slide. You see that a lot of things need to be considered here: the coordination, the production,... the information flow is particularly important from a BIS perspective. e-SCM is electronic supply chain management, that will be the collaborative use, webbased technology predominantly for conducting SCM activities.

**Slide 5:** In the year 2000, lots of predictions were being made by companies like Gartner. Some of their predictions can be seen on the slide. SCM has evolved a lot, but not all industries have been equally affected by B2B ecommerce. The ones that have been very sophisticated at doing B2B e-business are especially the computer industry, automotive, chemical and industrial equipment industries. Those are the ones where a competitive advantage has been created by means of e-business in a SCM setting. We’re going to discuss some examples of these in the remaining of this chapter.

**Slide 6:** If you look at how a traditional supply chain looked, it was something that was very linear and structured. It was what we call point-to-point. We had tier 3, 2, 1 suppliers, the manufacturer, the distributer and the retailer who sold the goods directly to the end consumer. It was highly structured and it was linear. It was not very flexible/agile. The chain was something that was linear, point-to-point and tight coupling, so very intense coupled relationships.

**Slide 7:** With the Internet, we can aim for loose coupling. We’re going to set up a very flexible architecture, according to a plug-and-play philosophy, where you have a distributor, retailers, logistic providers, manufacturers, maybe some virtual manufacturers, some contract manufacturers, some suppliers (tier 1, 2, 3). The goal is to set up a very flexible architecture, such that if you want to unplug one particular party, such that you can do it without a problem. And if you want to plug in another party, such that you can do that without any problem. In order to accomplish this very ambitious goal, you need to have the appropriate underlying ICT technology. So our information systems should be versatile, flexible, agile, so that we can accomplish this aim of the future Internet-driven supply network such that we can accomplish this aim of setting up a plug-and-play architecture. So, we should aim at loosely coupled relationships.

* By doing so, our (linear) chains will evolve into collaborative networks where different partners can collaborate in a very flexible way.
* That will also allow us to come up with pure play or virtual companies (which are only present on the Internet and which collaborate with a lot of other partners, also only available on the Internet in order to conduct their main business).
* However, only a few successful architectures are available of this architecture example. Think about Dell, Amazon,... → Examples of setting up these collaborative networks in a particular business setting.

**B2B technologies**

**Slide 9:** Electronic Data Interchange is legacy, that means it’s software that’s used in the past. That does not mean that it’s not relevant anymore, because some companies are still using EDI technology nowadays. In EDI, you’re going to transfer electronic data from one particular firm to another. This EDI message will then be processed by the receiver’s computer system without the need for human interpretation, so it can be fully automated. We do not only need software to read the EDI messages, but also hardware to pass on the message from one computer system to another. We need value-added networks, networks that are going to add value in terms of bridging a gap or making a connection between one firm and another firm. There can be all kinds of international standards, lots are proposed, like UN/EDIFACT. These are outdated standards, because they implement an outdated technology! EDI was used since the 1980s to automate routine transactions between established trading partners, it was long-term. We had to make a point-to-point connection between two firms, we had to have a cable available between two firms and it was tightly coupled, not loosely coupled. This is still begin used quite widely in B2B integration practice.

**Slide 10:** Here you can see how it goes: you have two computers, you have a supplier and a manufacturer, which are going to exchange all kinds of messages like purchase orders, payments,... All these messages can be standardized as EDI messages. EDI is basically a syntax which will allow you to come up with a purchase order, a payment,... There have been large EDI groups, implementing EDI in particular settings, like SWIFT.

**Slide 11:** Here you can see how it works. Let’s say we have 3 companies and we want them to work together in terms of EDI. Each of these companies uses their own proprietary software, written in old programming languages. Company A cannot talk to company B unless they come up with a common language. That common language here is EDI. If A puts a purchase company with company B, that order is first specified in A’s proprietary language. It is then transformed to an EDI message, which is sent along a network to company B and the incoming EDI message is then translated into the language that B understands. What you’re doing each time is translating an EDI message to a particular language, or you translate a language-specific message to an EDImessage. You’re translating all the time. Middleware is a piece of software that is going to do that translation for you. It’s software that translates from one language to another.

**Slide 12:** Example of an EDI message. Looks a lot like XML. Also has tags like NAD. It has all kinds of delimiters which are all part of the system, which can be used to exchange information in a standardized way.

**Slide 13:** Benefits:

⊕ Speed of transaction: you can very quickly send a purchase order from one organization to another.

⊕ Reduced transaction costs: you don’t have to call the organization or pay a receptionist, which are answering the purchase order calls. You can all automate it and send purchase orders along via EDI.

⊕ Less errors: EDI messages have been standardized, with fixed delimiters, so there are less errors.

⊕ Just-in-time support: lower inventory costs: because of the efficiency of the communication flow, the inventory can also decrease which will imply lower inventory costs.

By investing in EDI, companies can gain in competitive edge over other companies in the sector. But it’s not always as sunny as it looks, because the advantage for the supplier and customer depends heavily on the balance of power between supplier and customer.

**Slide 14:** Imagine that you’re a very small firm and you’re setting up an EDI collaboration with a bigger firm. Imagine that you’re a seller of pancakes to a big supermarket. You’re a very small firm, an SME (small and medium sized enterprise) and you want to set up an EDI collaboration with a bigger firm. Once the SME has made the investment in the EDI infrastructure, then actually can start abusing the balance of power, because now that it knows that the SME has made a significant investment into the appropriate hardware and software infrastructure, it can try and push the price downwards for that SME such that it can have those pancakes at a lower price. Because of the long-term, point-to-point and tightly coupled relationship, the big firm can make abuse of the balance of power and it can force the small firm to decrease its prices since it has made a significant investment into the EDI technology. This is the risk of being locked in, of being too focused on your ICT architecture, on one particular customer. If you’re too focused on one particular customer, you start to become too dependent of the customer and the customer can lock you in. He can force you to decrease your prices, knowing that you’re not likely to abandon your investment in the EDI architecture so, knowing that they have the power to reduce their prices. This is the idea of vendor lock-in. Because of the high initial investment, this can be particularly an issue for SMEs. These are the disadvantages of EDI that are originating from the fact that it’s long-term, point-to-point and a tightly-coupled relationship that is established, so it’s less suitable to be used in a networked, collaborative environment. That’s why it would be good if we could shift away from this private communication networks and proprietary hardware/software towards using Internet technologies.

**Slide 15:** HTML is one of those Internet technologies. HTML stands for HyperText Markup Language. It’s the language that specifies representation of the content of pages on the Internet. We’re going to specify what text should be bold, italic,... All of that will be represented in HTML. It’s not contentoriented, but presentation-oriented. And because it’s presentation-oriented, HTML is not suitable to do e-business.

**Slide 16:** The newest version of HTML, HTML 5, has a very simplified syntax, it is backwards compatible towards previous versions of HTML, it has various new elements and attributes, it has CSS for maintaining consistency in terms of formatting guidelines between different pages of a website, but it also puts an increased emphasis on DOM and scripting languages. It just remains a presentation-oriented standard, not focused on the content itself.

**Slide 17:** That’s why it’s not very suitable to be used in an e-business environment. Presentation-oriented, it has no structural or semantic orientation. It does not allow me to do content-based querying. It cannot be effectively be searched by process and business applications.

**Slide 18:** That’s why XML is a lot more efficient to do this, because it will focus on the content. We will not make use of embedded formatting information. We just specify the content. The formatting itself will be done by means of style sheets (what item should be represented in what format (like italics, bold,...)?). XML is a meta-markup language, it doesn’t have a fixed set of tags and groups of users can define their own tags for a particular domain or purpose. So business users can define their own tags for their purpose. In banking for example, you can come up with your own tags which are relevant in a banking setting. Those definitions are referred to as schemes, they are used to define your tags and define a structure for a collection of XML document. Those XML documents can then serve as a uniform data exchange format between B2B applications. All these documents that you need in a B2B environment can be appropriately specified using XML tags.

**Slide 19:** Here you can see an example of a catalog entry in XML. It’s a notebook specification, so you can see that the tags are not representing the format but the product. If I now want to query this catalog entry and search for the price of a particular notebook, I can very easily do that. You can also ask for the currency and transform it to another currency. XML is a very suitable format to be used in B2B e-business. Using XML, we can specify the content of documents and it’s very easy to use this in B2B exchanges.

**Slide 20:** Now, companies can start exchanging XML documents, which can be very efficiently understood and parsed. The XML document can be processed fully automatically, there exist all kinds of XML processing tools that will allow us to do this.

**Slide 21:** XML is only a data format, it’s not going to tell you something about the processes of actually putting a purchase order within a particular firm for example. It only specifies the data. So what about the business process and the messaging? When you set up a purchase transaction, typically there’s a set of messages that are going to be exchanged: you put a purchase order of the supplier, you get a confirmation of the purchase order, then the supplier is going to send the goods and you send a message that you received the goods,... → We have a set of messages that are being exchanged between the customer and the manufacturer. XML will only specify the content of those messages, but if we also want to structure the process model, we need other standards: B2B e-business frameworks. There are 2 standards here: ebXML and RosettaNet. They are two standards which are going to describe the process steps in setting up or working out a purchase transaction. They’re not data standards, they’re process standards. XML is a data standard. They will help you doing the steps in something particular, like issuing a purchase order. All of that is then automated in a web environment with web services and the idea here is to establish that plug-and-play architecture. That we defined earlier on.

**Slide 22:** Let’s now have a look at the classification of B2B models. Roughly speaking, you can make a distinction between a sell-side B2B and a buy-side B2B. In a sell-side B2B, you have one seller and many buyers. Usually, the seller is going to provide a shopping website with an e-catalog that will allow the buyers to log on to the catalog, specify the products they want and order them. The seller can also sell via auctions, like on eBay. Buyers are then going to shop directly on the seller’s website, configure the products and order them. In a buy-side B2B, we have one buyer and multiple sellers. Usually, the buyer is going to aggregate multiple sellers, catalogs and trade using reverse auctions and/or negotiation. Remember, reverse auction is the auction where the buyer is interested in stationary material like pens,.. And here’s she’s going to ask for a quote for that particular order with particular sellers and she’s then going to order the cheapest probably. → Downward pressure on the price. Typically, the front-end should be integrated with the company’s back-end applications, using the idea of enterprise application integration.

**Slide 23:** Example of a sell-side model, operating in Hong Kong, supplying in an extensive range of office products.

**Slide 24:** Example of a buy-side model: click2procure. If you’re a firm and you think you have an interesting product to offer to Siemens, you can log on to this buy-side B2B site, create a profile for your firm and specify what products or services you can offer to Siemens. They will have all that information loaded into their databases and screen whether you have something interesting to offer.

**Slide 25:** E-marketplaces are public electronic market places where many buyers and many sellers are going to connect: virtual market place, available on the Internet. It will offer online services, like catalog management. Those e-marketplaces can become very transparent and competitive because all the information is available on the e-marketplace, it can very much encourage competitive bidding. That’s why many suppliers are often reluctant in participating in e-marketplaces because they know that by participating, they are destroying each other.

**Slide 26:** Covisint: very well-known car manufacturers. They wanted to create a web-based platform which allows participants to work across computer platforms, operating systems without requiring participating companies to invest in special systems.

**Slide 27:** It was not a success: there was a split mission: force prices downwards, to create very efficient competition between different suppliers. Basically, what the suppliers were doing was destroying one another. That created a lot of distrust between partners. Also, the Covisint initiative was one of big car manufacturers and there was a lack of leadership because no-one would allow another car manufacturer to take leadership of the e-marketplace. This created reluctance of the suppliers, they were not willing to participate in the e-marketplace, because they knew that by doing so, that they were going to force each other out of the business. Then we have the rival technology providers.

**Slide 28:** Covisint was not really a success in terms of an e-marketplace. What was more of a success in the automobile sector was SupplyOn. They didn’t want to drive prices down and create competition, it was set up to be a lot more supplier-friendly. It was established in the mid-2000s by a few suppliers in the automobile industry.

**Slide 29:** Even nowadays, there are more than 1000 automobile industry suppliers which use this e-marketplace daily. They provide all kinds of facilities, for the seller to specify its community profile, to provide technology and know-how, its manufacturing capabilities,... For the buyer to identify potential suppliers,... Of course it also has a sourcing manager, a piece of software which will allow you to find the source, to find a supplier for your particular request. You can then create and send an RfQ (Request for Quote), the seller will then send a quotation. This can then be automated in a bidding process, where quotations can be compared in terms of their price, delivery terms,... Reviews are also possible. Bidding is also part of the functionality which is typically provided by SupplyOn.

**Slide 30:** In order to be fully integrated with the existing legacy applications, the SupplyOn marketplace also has an interface with EDI, which will allow us to come up with EDI-based messages such that we can set up EDI communications and so on. It also has XML interfaces.

**Slide 31:** Here you can see this again: the buy-side and sell-side company by means of the SupplyOn WebEDI, we can just send purchase orders along, prototype orders, advanced shipping notifications,... All of that can be passed along through the e-marketplace.

**Slide 32 & 33:** Types of e-marketplaces: we can categorize according to types of transactions and types of materials traded:

* Types of transactions:
  + Spot buying: you’re buying on the spot.
  + Strategic (systematic) sourcing.

→ Big difference between both.

* Types of materials traded:
  + Direct materials: materials that are directly used in the production process.
  + Indirect materials.
  + MRO.
* Direction of trade:
  + Vertical: it’s going to provide functionality across the different business units within a particular industry or industry segment.
  + Horizontal: items which are necessary in a variety of different industries.
* Degree of openness.

**Slide 34:** Examples.

**Slide 35:** OneAero: marketplace that connects airlines, OEMs (Original Equipment Manufacturers).

**Slide 38:**

* E-marketplaces are publicly available and bring together potentially thousands of sellers and buyers in a digital marketplace.
* Private industrial networks: it’s shielded from the outside world. It’s collaborative commerce, whereby a set of partners, based upon their relationships, which are private, are going to set up collaborative commerce.

**Slide 39:** Example of a PIN: this is a set of businesses which set up a private collaboration, hidden from the outside world, you can’t just join this PIN, unless you’re begin invited by the partners.

**Slide 40:** The goal of these PINs is to deal with the bullwhip effect. In the linear model, each one needs to decide how much inventory they are going to keep, it is usually determined by the forecast of the needs downstream. The inventory for the retailer, depends on the forecast consumer demand, so the retailer is going to make use of business intelligence, business analytics or data mining to project or predict customer demand. The retailer is then going to make sure that (s)he has enough products in order to be able to sufficiently handle the customer demand. Typically, you’re going to see that they’re going to keep a little bit of excess inventory. What you see is that a small variation in customer demand is going to lead up to a bigger variation in the retailer inventory, because the retailer wants to make sure that (s)he has enough products or services to deal with the customer demand. The distributor, in turn, will project using BI the retailer demand. (S)he will also make sure that (s)he has enough products in store to supply the retailer demand. You also see that a small fluctuation in the retailer demand, will cause a bigger fluctuation in the distributor inventory level. What you see is that the variability in the inventory increases as you go up the supply chain. So a small variability in customer demand, will create a bigger variability in inventory as you climb up the supply chain. Bullwhip effect: you whip it and you get a bigger fluctuation as you approach the end of the whip.

**Slide 41:** One way of dealing with this, is by using a technology which is called Vendor Managed Inventory. So, now, as a supplier, you can get access to the inventory of the manufacturer, you get direct access to it. The supplier or the seller will then be able to optimize production planning and overall supply chain efficiency, which is one way of countering the bullwhip effect. The buyer needs to open up its inventory to the supplier, which means that he’s losing some control, but in return for that, he’ll get cost savings which can result in a more competitive end-consumer price. So, VMI means that a seller gets access to the inventory levels of the buyers and that way, the seller can optimize the production planning and that can create cost savings.

**Slide 42:** Sometimes, things can go wrong as well: Nike was using BI to do demand forecasting. Because the forecasting didn’t work very well, the software caused the stock value to plummet 9 months later. They ended up ordering the wrong type of shoes. The analytics software had indicated that those shoes were going to be in high demand. They were confronted with this huge amount of shoe stock they could not sell. The reason it did not work well was because the analytical models were not well estimated/designed.

**Slide 43:** Walmart: doing lots of investments in this field, they developed collaborative commerce in the late 1980s. In the 1990s, they introduced Retail Link, which is a kind of PIN. They used that system and upgraded it to a CPFR system. ⇒ All those developments are used a lot by these big firms.

**Online advertising**

**Slide 45:** We’re going to do this advertising on the Internet. We have a whole range, different means to do this online advertising.

**Slide 46:** Here you can see the US online advertising between 1996 and 2012. It keeps on increasing, with two bumps. The overall trend is increasing.

**Slide 47:** Here you can see the online advertising in Western-Europe, between 2010 and 2014. You can see it’s increasing as well.

**Slide 48:** There’s different types of online advertising as well. We will extensively comment on them.

**Slide 50:** E-mail advertising is not very effective because, for many of us, if we receive ads by e-mail, our spam filter will remove them.

**Slide 51:** The publisher is a website, he’s going to provide content, sometimes for free, and services, like e-mail,... This works best when the volume of viewer traffic is large or highly specialized (particular audience).

**Slide 52:** We can also aim for different levels of intrusiveness:

**Low:** Customers just have to click on a banner and they only do it when they want. It’s not bothering you a lot.

**High:** Window which appears in front of another window. It can get very annoying.

**Challenge:** You want to make sure that people see your message, but you also want to make sure that they do not get frustrated with it.

**Slide 53:**

* Text ads,... can be static (just show information) or contain rich media (videos, games,...)
* Pop-up: pops up in front of the website you’re visiting. Pop-under: appears below another window, you don’t see it initially, but when you close the main window, you’ll see that another window is still open.

You can stop them using pop-up stoppers.

**Slide 55:** Example.

**Slide 56:** We also have interstitials, which is a full page which is loaded before the actual page that you want to visit. This will last for a few seconds and then you’re redirected to the intended page.

**Slide 58:** Layer adds: adds that scroll down as the visitor scrolls down the page. When advertisers want quick attention and action. They’re usually quite expensive.

**Slide 59:** You have an article and the layer ads software is going to advertise it and it will see that it’s about cars and will then show an ad for Toyota. Not a very appropriate add in this case. Depending on where you are in the text, other types of layer ads could appear.

**Slide 60:** Many of us suffer from Internet Ad Avoidance. We all have favorite websites, and eventually we’ll know which ads are shown, so we’ll ignore them. Banners typically have very low click-through rates. Users typically also suffer from banner-blindness.

**Slide 61:** Search-based advertising:

**Unpaid:** You’re not going to pay a search engine like Google or Yahoo!, but you’re going to make sure that your website has been well-designed such that it can be easily detected by Google. You want to rank highly in search results. Make sure the appropriate content is on your websites and that the correct websites link to you.

**Paid:** Pay the search engine to include a link to your website.

**Slide 62:** With Google, this can all be very nicely implemented.

**Slide 63:** Links to websites that are sponsored links: paying Google to include a link to their websites.

**Slide 64:** SEO is unpaid, you just optimize your website to improve your ranking in the search results. That means that you have to make it easy for search bot to find your site. We need to keep it easy for search bots to find our websites and we need to specify and provide sufficient text content. The URL should also be simple and easy to understand.

Link farm: a malicious way of improving your results in a SEO. It’s a configuration whereby you’re going to have a lot of (fake) websites linking to your website, to make your website more important in the search engine results. It’s trying to fool Google. It can be detected by Google and may result in Google not including your website in the search results.

**Slide 65:** SEO can be very neatly done by specifying the right information in the robots.txt file. It’s a file that every website should have and it’s going to direct search bots through your website. Sometimes you don’t want certain pages of your websites to be indexed, to be known, you can specify that in your robots.txt file.

Crawl delay: make sure that after each visit of a webpage, it needs to wait a couple of minutes before it can look at a next page. You do that to not overload your website. You do not want the Google bot to overload your website, while other users are also inspecting your website. By means of the crawl delay, you can specify that a bot needs to wait a couple of minutes before it can visit another page.

Overview of website hierarchy with information for the search bot.

**Slide 66:** robots.txt file from Twitter.

**Slide 67:** Change frequency of the page: if a high change frequency, it means that it is frequently being changed, and the search bot needs to check it more often. If low change frequency, the search bot can work more efficiently by skipping the page.

**Slide 68:** Example of sitemap from Amazon.

**Slide 69:** For video content, there’s more information that you can give that can facilitate the crawling by the bots. You can specify the content of the video and the title, where the thumbnail is located,...

**Slide 70:** Google works according to the PageRank algorithm. You need content and lots of incoming references. If it has a lot of incoming references, it means that it has something useful to say. It should not only have a lot of incoming references, but also a lot of references from other important pages. This is the democratic nature of the Internet. C is very important, despite it only having one incoming link. The reason that it’s so important, is because it has an incoming link from B, which is also very important.

**Slide 71:** How Google PageRank works. The PageRank of a particular page depends on the PageRank of other pages that connect to that particular page.

**Slide 72:** Start by applying page rank 1 to every website and then continue iteratively until there’s stability.

**Slide 73:** You can abuse this Google PageRank and come up with a Google bomb. As the Americans invaded Iraq, many people did not like this (opposed the idea of using violence). What they did was starting a hate community and they massively linked to the White House using the keyword ”failure”. If you start to massively linking with a word, then Google will detect that. If it occurs, you can report it to Google and they will make sure that the Google bomb is removed.

**Slide 74:** A very popular ad tools is Google AdWords. The ranking of the ad is based on cost-per-click-bidding campaigns. You can bid for a link to your website to be included on the ”sponsored links” section. You can do that using pay per click (you only pay when someone clicks on your ad) or using the Max CPC. The pay per click is higher for A than B and C.

**Slide 75:** You can specify a lot of things when you advertise. You can really customize your ads to a region or customer population.

**Slide 77:** Local ads will be included in Google Maps and will show the business and the location.

**Slide 78:** An ad network is a very sophisticated way of doing advertising. Imagine you set up your own website and you’re a soccer fan and you want to set up your own website about your team. You could try and gain some money from your site by using an ad network to do the advertising on your website. One of those is Google AdSense: you can let Google put advertisements based on your website. It will look what your website is about and put all kind of advertisements there that are connected to the content on your website. Google will do the advertising for you. The ad network and the publisher of the website each get their share of the revenue of the ad income.

**Slide 79:** Pixel targeting: you’re only targeting a particular part of the computer screen.

**Slide 80:** Website on the weather forecasting and ads that have been put there by Google AdSense.

**Slide 81:** Google AdSense makes use of different types of cost schemes: CPC, RMP or CPA.

**Slide 83:** Social advertising: the way Facebook does it. You have to be careful with that, because FB has invaded privacy quite substantially in the past.

**Slide 85:** Ads are being put there depending on the profile of the person, the friends and the actions of the friends.

**Slide 88:** Facebook has the idea of the open graph whereby it provides an API that programmers/website designers of other websites can use in order to interact with Facebook for doing advertisers. It allows developers to connect their product to the Facebook’s social network. That way, your activities are propagated to your friends so as to influence them.

**Slide 90:** If you like Skyfall (on the right of the screen), all that information is passed on to Facebook, saying that you liked the movie.

**Slide 91:** All kinds of additional information you can pass along.

**Slide 93:** Very important Internet service. You can reach customers as they watch videos.

**Slide 95:** Marktrock can be sponsored by a company to set up the website.

**Slide 96:** Groupon,... → Register first and provide information about yourself and then it will target coupons based on your interests.

**Slide 97:** Dan Brown gave The Da Vinci Code to 10000 readers for free. It’s a marketing technique that’s going to use social networks to spread a message.

**Slide 98:** People sent it to their friends → went viral.

**Slide 99 ev:** Different pricing models:

**CPM:** Allows you to brand a business’s name, you buy a guaranteed number of appearances of your ad.

**CPC:** The advertiser is only going to pay once per click.

**CPA:** You only pay when a particular (predefined) action has occurred.

The most frequently used models are CPM and CPC.

**Slide 102:** This creates a lot of possibilities for click fraud. As a rival, you can deplete a competitor’s budget by clicking on his website ”every now and then”. If you want to earn money yourself, you can click on your own website.

**Slide 103:** There have been some cases as seen in the slides.

**Chapter 26**

# Les 21: 19/05/2015

Vandaag: tweede flipped classroom session.

## 26.1 Slides: 7D.E-business-part1-QuizSession-Student Version

**Liquidation vs market efficiency auctions uitleg:** Mogelijkheid die eauctions geven om die op verschillende manieren te organiseren. Die worden op 2 manieren gecategoriseerd.

**Liquidation:** Producten trachten te verkopen via dezelfde soorten kanalen/klanten en proberen de resterende producten te verkopen aan een lagere prijs.

**Market efficiency auctions:** Doordat je een e-auction gebruikt, kan de prijs voor die overstock nog relatief goed zijn en je bereikt een breed publiek. Het gaat hier vaak om vrij unieke producten en je wil een heel breed publiek bereiken om een markteffici¨ente prijs te bereiken.

Zijn 2 verschillende doelstellingen die je kan hebben om zo’n e-auction te organiseren.

**Amazon Mechanical Turk:**

* Human intelligence tasks: opdrachten die je kan laten uitvoeren/outsourcen via Amazon Mechanical Turk: kleine menselijke acties die je kan laten uitvoeren door mensen overheen heel de wereld. Voorbeeld daarrond: een van de vreogere PhD studenten was in NY bij een e-advertising bedrijf gaan werken. Zij zijn ge¨ınteresseerd in het niet-plaatsen van hun advertenties waar zij als bedrijf niet op willen staan. Link naar hier: om die classificatiealgoritmes te trainen, moet je een dataset opzetten, bv porno vs geen porno. Wat die onderzoeker gedaan heeft, is Amazon Mechanical Turk gebruiken om 100 000 websites te laten classificeren, manueel, en op die manier een correct label te krijgen. Zo kreeg hij een classificatielabel dat heel accuraat was en waarmee hij zijn model kon trainen.
* Crowdsourcing: gebruik maken van het brede publiek en zich te laten engageren om te doen wat jij wil.
* Requesters/Turkers: degenen die services vragen en aanbieden.
* Marketplace for work
* Commissie: bij AMT: 10%.

**Wisdom of crowds uitleg:** Heel algemene term die niet noodzakelijk alleen bij e-business hoort. Gebruik maken van een ruim publiek om kennis te produceren. Wikipedia is hier een vb van. Ook in de gokwereld.

**True or False vragen, ronde 1:** Vragen: zie Figuur 26.1.

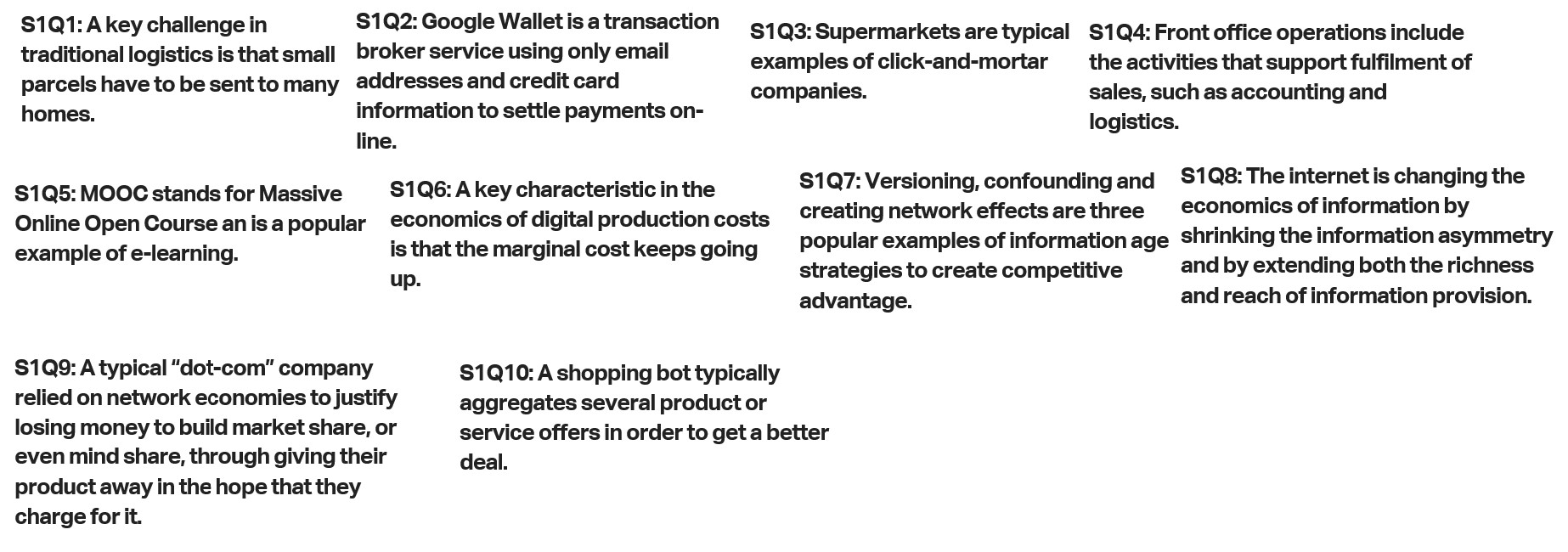


Figure 26.1: Vragen van True/False quiz 1.

**Q1:** False: in traditionele logistics gaat het net niet om kleine pakketjes, maar net om relatief grote pakketten. Dit is de defnitie van e-logistics, daar gaat het om het feit dat je kleine pakketjes (Zalando,...) thuis gaat afleveren.

**Q2:** False: Het is inderdaad via smartphones, maar dit gaat over PayPal. Zij zijn een transaction broker die werken met dat model. Google Wallet werkt ook met een applicatie op de smartphone om die betalingen te kunnen uitvoeren.

**Q3:** True: bedrijf dat zowel in de fysieke wereld als online verkoopt. Supermarkten vandaag de dag verkopen zowel online als in het echt.

**Q4:** False: dit zijn back-office operations.

**Q5:** True

**Q6:** False: margin costs zijn constant

**Q7:** True: vbn van hoe je competitive advantages cre¨eert

**Q8:** True

**Q9:** True

**Q10:** False: de uitleg is geen shopping bot maar een aggregator.

**Richness vs reach uitleg:** Bv: het feit dat je veel meer mensen kan bereiken via het Internet dan vroeger (reach). Richness: ruimer concept. Yelp: website die reviews van horeca geeft en daar scores aan geeft, reviews publiceert,... → Vb van hoe informatie veel rijker wordt en veel ruimer: je kan opinies van mensen in real-time krijgen. Ook alleen de waarde van informatie al. Al die zaken kunnen veel rijker aangevuld worden. Yelp is een vb van hoe het Internet toelaat om veel rijke infobronnen te publiceren. Het Internet laat toe om zowel de richness als de reach omhoog te brengen.

**Constante marginale kosten uitleg:** Ongeacht hoeveel je gaat produceren, de totale kosten gaan niet sterker stijgen met de productie van 1 extra eenheid. Wil niet zeggen dat je geen extra kosten meer doet! De totale kostencurve gaat nog steeds omhoog, maar de kosten stijgen niet sterker dan lineair, niet sterker dan de eenheidskost per product.

**Slide 37:** De slide van e-commerce is vrij belangrijk (**Slide 7** staat er op de slide). Belangrijk van dat goed te kennen en je moet in staat zijn om dat uit te leggen. Vraag: wat is browser sharing? Support vragen en aan de hand van browser sharing in real-time jouw configuratie maken. Ze kunnen jouw computer overnemen en bepaalde opties selecteren of extra uitleg geven. Je gaat de browser van een (potenti¨ele) klant overnemen. Post-sale: aantonen hoe een product werkt.

## 26.2 Slides: 7E.E-business-part2-QuizSession-StudentVersion

De meeste content komt in deze web lecture aan bod.

**EDI:**

* Middleware: programmatuur om bedrijven met elkaar in verbinding te kunnen laten staan via het Internet.
* Electronic Data Interchange.
* Point-to-point: geen centrale hub die die EDI messages verstuurt.
* Long-term: je gebruikt EDI met het oog om langetermijnrelaties aan te gaan met het ander bedrijf.
* Vendor lock-in: te afhankelijk worden: als bedrijf investeer je in technologie. Die EDI technologie is specifiek voor die relatie: je cre¨eert een shift in the balance of power. Je kan onder druk gezet worden door het feit dat je ge¨ınvesteerd hebt in die technologie.

**B2B models:**

* Sell-side B2B: 1 verkoper en meerdere verkopers.

Buy-side B2B: 1 koper en meerdere verkopers.

* E-marketplace: eBay voor bedrijven die een B2B marktplaats oprichten.

Je hoeft geen lid te zijn.

* PIN
* Vendor managed inventory: manier om in e-supplychain het bullwhipeffect onder controle te krijgen. Andere bedrijven krijgen toegang tot jouw inventory.

**Display advertising:**

* Banners: horizontale reclameblokken op een website.
* Pop-up/pop-under:
  + Pop-up: window dat bovenop de website verschijnt. Nieuw browservenster dat opent voor de huidige browser.
  + Pop-under: onder uw huidig venster. Is iets minder invasief.
* Skyscrapers: advertenties links/rechts van een website. Verticaal geori¨enteerd.
* Layer ad: extra laag bovenop een huidige pagina die reclame toont die soms nog toelaat om een deel van de website waarop je surft te laten zien.
* Interstitials: soort van layer ad, alleen palmt deze een volledige pagina in en laat deze niet toe dat je nog iets doet op de website zelf. Geen aparte browser, maar een layer bovenop de website die de volledige grootte van de website inpalmt en niet meer toont wat de ”echte” site toont. Je kan niet verdersurfen tenzij je de boodschap toeklikt of lang genoeg wacht.

**SEO:**

* Search engine optimization.
* Robots.txt: info voor crawlers.
* Unpaid: zelf je ranking op Google/Yahoo/... bepalen.
* Search bot: software dat op een automatische manier websites gaat opzoeken en die pagina’s in kaart gaat brengen.
* Sitemap: bestand dat aangeeft hoe de verschillende pagina’s van jouw website aan elkaar gelinkt zijn.

**PageRank:**

* Google
* Adjacency matrix
* Iterative computation
* Google bomb: vI’m feeling lucky button. Die bestaat ondertussen niet meer. Je ging niet meer naar zoekresultaten, maar naar het eerste resultaat. Misbruik maken van de functionaliteit van die button. Is nu veranderd naar Google Instant: laat toe om op basis van een bepaald keyword te linken naar een bepaalde service.
* Internet democracy: vnetneutraliteit. Pagerank was oorspronkelijk heel nobel in zijn doelstelling: democratische manier om de belangrijkheid en relevantie van websites te schatten. Websites stemden voor mekaar door links naar elkaar te leggen. Soort van vote van een website naar een andere. Pagerank be¨ınvloedt die democratie steeds meer: link farming proberen te traceren.

**Adwords:**

* Google
* Pay per click
* Search-based advertising
* Maximum cost-per-click
* Locality

**Click fraud uitleg:** Niet correct op slides: ads van een publisher: bedrijven die websites hosten, die kunnen geld verdienen door als publisher zelf te klikken op de ads die je publiceert. Je klikt dus op ads van je klanten. Staat tegenover competitor click fraud: concurrenten uit de markt drijven door actief te klikken op links naar de concurrent.

**Ad network:**

* Google AdSense (AOL is een ander vb.)
* Intermediary
* Targeting
* Quality control: controle over de kwaliteit van de websites waarop jouw ads gehost zullen worden.
* Ad rotation: ad network is in staat om op hele set van websites de advertentieruimte dynamisch te gaan invullen.

**Facebook OpenGraph:**

* Social advertising
* Meta property tags: Facebook toevoegen aan jouw website.
* Notifications
* API

User driven: jij doet als gebruiker zelf zaken, actief, en adverteerders maken daar gebruik van om te adverteren.

**True/False: vragen, ronde 2:** Vragen: zie Figuur 26.2.

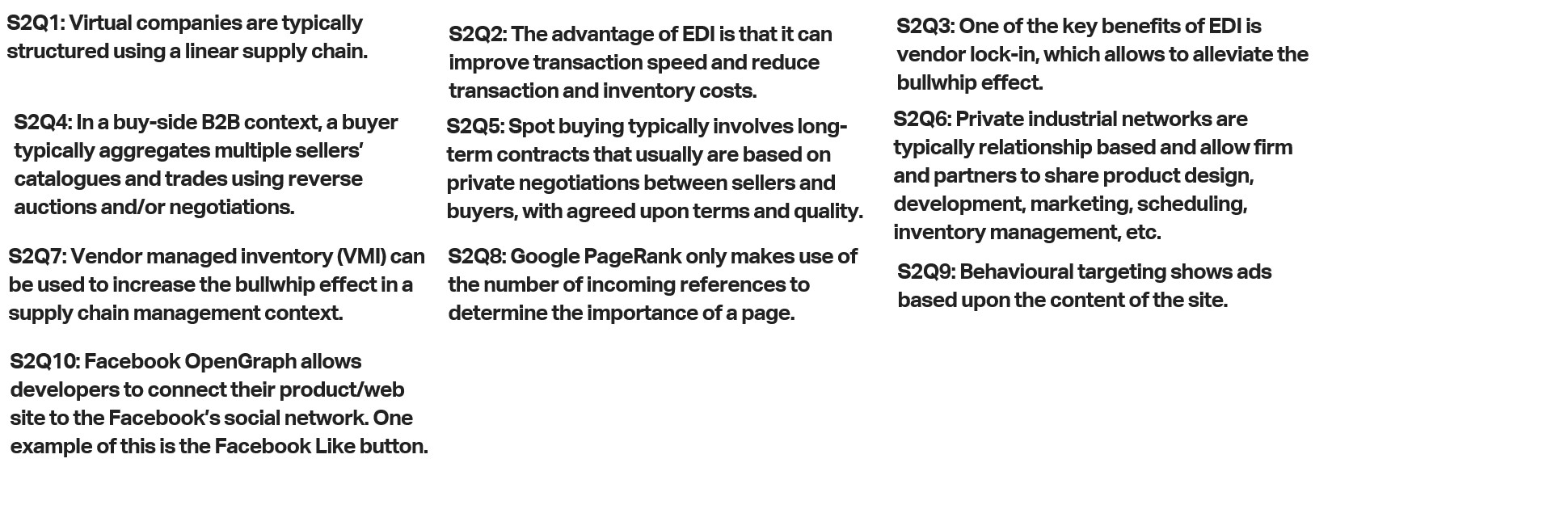


Figure 26.2: Vragen van True/False quiz 1.

**Q1:** True

**Q2:** True

**Q3:** False

**Q4:** False

**Q5:** False, is strategic sourcing.

**Q6:** True

**Q7:** False

**Q8:** False

**Q9:** False, is contextual advertising

**Q10:** True

**Chapter 27**

# Lecture: Business Information Systems 9-3: E-business vLes 22

## 27.1 Slides: 7C.E-business-part3

### 27.1.1 Video: Business Information Systems 9-3: E-business

**Slide 1:** Web analytics is also referred to as web intelligence.

**Slide 2:** It comes down to the usage of advanced analysis techniques applied to web data. It is also referred to as web mining. 3 types of web mining:

* Web usage mining or click stream analysis: it refers to how users are going to use your website: what pages are they typically visiting as part of their visit? What are the pages that are usually visited together? This is very useful information, because it allows you to understand how users are going to use your website.
* Web content mining: what Google does: analyze the content of your website to see what it is all about. Here, you study information and knowledge from webpage contents. This is very important for information retrieval and extraction, also for automatic document categorization, to see what documents are all about.
* Web structure mining: studies & analyzes the structure between websites. This will be done by looking at what website links to what other website, or what page links to another page. By doing so, you can find communities of interest, which relate to a similar topic/concept.

**Slide 3:** Goals of web analytics:

Improve website design: if your web analytics exercise tells you that two pages are often visited together, but there’s not link in between, you can use that information by making sure that there’s a link from one webpage to the other webpage. It will also allow you to see how users or segments of users are going to use your website: what pages do they visit? So you can customize it to individuals users.

* Measure the effectiveness of marketing strategies, SEM (Search Engine Marketing) (SEO (Search Engine Optimization)/PPO) activities and advertising campaigns: make sure that your website ranks high in the Google Search results for particular keywords. You’re going to do that without paying Google for it. In Pay Per Click, you’re going to pay Google in order to include a link to your website. One example of PPC is Google AdWords.
  + Monitor the effect of a new search engine strategy on traffic, conversion rate,...
  + Landing page optimization: first page that is seen during a web page visit. You can try out different variations of a landing page, each tailored to the needs of segments of particular users and see what landing page variation is most attractive for particular groups of customers.
* Personalization: personalize the webpages to the individual user needs. If someone logs onto Amazon, that person will see that the page is customized upon previous purchases.

**Slide 4:** If you want to do web data mining, you need data. Data can originate from a variety of different sources:

* Server logs: huge documents that are stored on the web server. Huge work documents in which information about web visits will be stored. The data is captured at the server side and every request for a page is going to generate an entry in a web server log file. This web server log file is a very important piece of information that can be parsed or processed on a fixed schedule basis to provide useful information. If I would log on to Amazon from this particular computer, then information about my visit will be stored in the web server log file maintained by the Amazon web server.

**Slide 5:** What information is captured?

* IP address of this particular machine: it can always be mapped to a geographical location. So Amazon will see that someone in Leuven is logging on to their website. This is very useful information because that way Amazon can see from what geographical regions it attracts visitors.
* Remote log name and user name: those will typically be blank.
* Date and time: very useful because that way they can see at what time users typically visit their website. If you launch a marketing campaign, you can check if the amount of users increases the day after.
* HTTP request method & resource requested: what page is being requested. Very useful because that will allow us to see what the most and least popular pages are.
* HTTP status code: indicates whether the request for the page was successful or not.
* Number of bytes transferred.
* Referrer: the website the user was on prior to the visit. It tells you something about your incoming traffic: where do your users come from. If the referrer is Google, then they will also send along the query string: the search term that was entered in Google. That’s very useful information, because it tells you what key words are going to lead to your website. If you want to invest in SEO or PPC, this can be very useful information. If you see that a lot of search terms include the words ”buy” and ”wine” then you know you won’t have to invest for those words in a PPC campaign because they already lead to your website.
* Browser and platform.

→ Lots of interesting information stored in the web server log about the visit of the particular user.

**Slide 6:** Cookies: typically, when you access a website (like Amazon), they will put a cookie on your machine which will typically contain a unique user ID. Every time you make a request to the Amazon website for a particular page, the cookie will be sent along. The website will then see that that particular user is asking for information. That way, the website will be able to look at visits and what pages are being requested from the same visitor. A cookie can contain all kinds of information, like a unique user ID along with other customized data, the domain and path from where it can be read. They can also expire. Cookies are subjected to a lot of regulations these days, especially in a privacy setting. By using cookies, a lot of information is gathered about your visit. So users should be informed about that. There’s regulations being used at European level, saying what information can and can’t be stored and also the regulation stipulates that the users have to be informed about the data collection that is taking place. That’s why, if you go to a website nowadays, you’ll typically need to approve whether the website can store cookies.

**Slide 7:** Page tagging: collecting information of your visit. This is clientside data collection. The user is going to request a page from the web server, say, from Amazon. They will then send the HTML code of the page and in that HTML code, there’s a JavaScript page tag that is typically stored on a data collection server, let’s say Google Analytics. The JS program is stored on Google Analytics and will be downloaded to your PC where it will be executed. Typically, that JS program is going to store a cookie, to identify you. All kinds of information about you and your visit is being gathered and sent along to the data collection server, Google Analytics. The website administrator can then have access to it and see all kinds of information on the usage of the site.

**Slide 8:** Very important for page views is that we have to define the contents of a page: what is a page? It’s not always easy to define this. It’s possible that you’re more interested in more aggregated level, not in too much detail. They might not be very useful because they don’t tell you anything about the user experience. Maybe the page views are up because the user was very frustrated because (s)he wasn’t able to find the information (s)he was looking for.

**Slide 9:** It’s a lot more interesting to look at visits or sessions. The example (index.html → products.html → ...) are pages visited in one session by one user. We need procedures for ”sessionization”: come up with the sessions or the visits. This is typically done based upon a set of heuristics, rules of thumb. IP-addresses for example. If in the web server log you find 2 page requests made by the same IP address in the same user agent, then this is already an indication that it’s probably part of the same visit. We can also look at cookies and user identifiers that are stored in cookies. If you see in the web server log that you have 2 page requests made by the same user ID, then this is a very likely indication that this is part of the same visit. The user ID is often sent along as URI parameters in the web page request, of the uniform resource indicator. Different software vendors selling web analytics software are going to do that differently: combine the information differently. That’s why it’s not that straightforward to reconcile different metrics from many different software vendors.

**Slide 10:** Once we have the information described above, we can look at visitors. You can have new visitors and return visitors. This is very useful for determining site affinity and loyalty.

**Slide 11:** You have ABCD, all referring to visitors and 2 periods. Then you can look at the number of unique visitors for those periods. New visitors are visitors that we have never seen before, so this is B and C in period 1. Return visitors are visitors that you have seen before, so in period 1, that’s A. All of this is checked by using cookies. This will never be perfect because people remove/block their cookies, so the measurements might not be accurate. All these measures are typically calculated during a particular period and we can’t just add up the numbers for these different periods.

**Slide 12:** A much more interesting question than the 3 at the top of the slide is how engaging our content is and what (doesn’t) work. It’s important not to misapply this. For example, on a support site, you don’t want people to spend too much time on there, because it probably means they can’t find what they are looking for.

**Slide 13:** Time on site cannot perfectly be measured. This is typically the case for many of these page metrics. Example: the web log states that the first page was viewed at 13.00 and a second page was requested at 13.02. then you know that the person spent 2 minutes on the first page. The third page was requested at 13.05, so you know that the user spent 3 minutes watching the second page. At that moment, you know the user spent 5 minutes on the site, but you don’t know how much time the user spent on the last page. The time on site is always underestimated! When the user visits only one page, then the metric becomes meaningless because you don’t know when the user left, so you have a time on site of zero.

**Slide 14:** If you want to plan a trip to Paris and you type it in in Google and you first click on a page of Paris Hilton. This will probably not be what you’re looking for, so you’re going to close it immediately and go to another search result. We call this a bounce: you click on a website, don’t find what you’re looking for and immediately leave. You can calculate the bounce rate of the site, which is the ratio of the single page view visits over the total number of visits. You can also calculate the bounce rate for a page, which is the single page view visits of that page over the total number of visits where that page was the entry page. Variations are possible.

**Slide 15:** The measures illustrated. A, B and E are pages.

Bounce rate of the site is 5/8, because you have 5 single page visits: Visit 2, 4, 5, 7, 8.

Bounce rate for page B: look at the visits where page B was the first page. This was visit 2, 3, 5, 7. 3/4 times, the user left immediately afterwards.

Page exit rate for page B: look at all sessions that include page B. This is visit 1, 2, 3, 5, 6, 7. 5 times, the user left immediately.

⇒ Measures that tell you something about the attractiveness of your website and web page.

**Slide 16:** Referrer information is also very important: tells you where your customers come from. If the visitor comes in via a search engine, you’ll also see the search engine key phrases. It’s very important to perform segmentation, once you have the referrer. You can segment based upon your referrers. You can look at your top 5 referrers and then you can look at the bounce rates for each of those top 5 rate referrers. If you have a high bounce rate from users coming from a particular website, it may mean that something might be wrong with the reference. You can also report on key phrases: the most popular search terms,... → Will allow you to see what search terms typically connect to your website.

**Slide 17:** Your top destinations or exit links are going to tell you where you’re sending traffic to. For example, if the site www.abc.com has a link to www.xyz.com, then you want to know that information about that click that links to www.xyz.com. When you click on that link, you will create an entry in the server log of www.xyz.com and the guys from www.abc.com cannot see that somebody is leaving their website to www.xyz.com. Ideally, the guys from www.abc.com want to see where they are sending their traffic or users to, because that gives you an idea about your top destinations or exit links. How to capture them?

External redirect: a link on www.abc.com is first going to redirect to an internal page of www.abc.com, which is then going to move the user forward to www.xyz.com. That way, an entry will be stored about the exit in the server log of www.abc.com and www.abc.com can see that it is sending traffic to www.xyz.com.

JavaScript-based exit tracking: if somebody clicks on the link to www.xyz.com, we can generate a JS event which will invoke a recording of the exit link stored on www.abc.com.

If you see that you redirect to another page a lot, you can try to get collaborations out of it: get traffic or money in return. Reciprocal links: have a link to your website on the website that is on your website.

**Slide 18:** Measuring outcomes using conversions: the visitor performs an action of your particular interest on your site. That action could be anything: downloading a pdf, ordering something,...

**Slide 19:** Conversion rate: the percentage of visits or of unique visitors for which we observed the event. You can, again, do it at the visitor level, but it’s also possible to record it during a particular time period.

**Slide 20:** Average visits or days to purchase: they are pan-session metrics: they span multiple sessions. It’s not for one session, but measured across different sessions.

**Slide 21:** Trends analysis: allows you to see if there’s a certain trend in certain metrics like bounce rates, conversion rates,... Once you see that trend, you can start wondering why. You can estimate time series, they’re statistical techniques that are going to give you projections in time.

**Slide 22:** This can be monitored using dashboards. They’re going to give you a web-based format of representing KPI’s. they’re typically concise, easy to understand, will give you real-time or periodic snapshots and include all kinds of indicators. You can personalize them, which means that everybody that is having access to the dashboard can indicate KPI’s (s)he is interested in. You can also provide links for further analyses. They’ll typically also allow for OLAP.

**Slide 23:** Example of a dashboard, implemented in SAS Web Analytics. You have a set of KPI’s, in the columns, like Metric. Every user can indicate in which KPI’s they’re interested in and they can personalize the dashboard.

**Slide 24:** You can make combinations of the techniques mentioned. That’s what’s so nice about web analytics: you can start to look for interesting patterns using segmentation. This is fully automated using the appropriate IT functionality.

**Slide 25:** Navigation analysis: studies how users navigate through you site.

Path analysis: checks what pages are frequently visited together.

Funnel: is going to look a pre-determined sequence and measure entry/abandonment at each stage. You want to see where they come from and where they leave to.

Page overlay/click density analysis: visualizes where users are going to click on your website. It can then be visualized as a heat map: it’s going to visit the whole webpage in a color-based way.

**Slide 26:** Example of a funnel in SAS Web Analytics. You can see the incoming traffic, based upon the top 5 referrers, you can see where traffic leaves after having seen the first page, how many people move on after having seen the second page. You can also see that you have new incoming traffic in the second page, how many leave and where they leave to. ⇒ Very useful information to be analyzed to further understand how users are going to use your webpage and how they’re going to navigate through the funnel.

**Slide 27:** Example: site overlay: visualizes where users click. You can see the plus signs on links and there you can see numbers about that link’s usage. This site overlay report can then be visualized as a heat map, which shows where users typically click.

**Slide 28:** Using web analytics, you can also see the impact on your SEM campaigns.

**Slide 29:** If you want to measure the SEO efforts, you can look at the inclusion ratio, the robot/crawl status reports, track inbound links or specialized tools like marketleap.

**Slide 30:** Other possibilities are Google Webmaster tools, track ranking for you top keywords/key phrases, see which keywords link to your most important pages and split and compare organic referrals versus PPC.

**Slide 31:** In order to measure PPC efforts, it’s very important that we can first identify pay per click because a user that comes in via Google can come in via SEO, but they can also come in via one of the links provided in the Google AdWords section. You have to make sure that you can make a distinction between both. You have to make sure that you tag the URIs with additional parameters. That will allow you to differentiate the incoming traffic from Google and see whether it comes in from SEO or via PPC. You also want to differentiate between bid terms and search terms: what was the term that you bid money for and what is the search term that Google connected to it? You will also get all kinds of additional data like ad impressions,...

**Slide 32:** It’s very interest to experiment and test. This will allow you to statistically compare the metric of interest, like the bounce rate.

**Slide 33:** Test different versions of a web page. If somebody clicks on a link to arrive at your page, you can split the traffic up into random samples. The webpages differ and then you can see if the bounce rates are different among the different versions of the website. Then you can statistically compare the metrics. This allows you to see what webpage is most attractive and you can also connect it to the user characteristics. That way, you really start to tailor you page to individual users.

**Slide 34:** More sophisticated: multivariate testing: test multiple things simultaneously on the webpage. You can play with these elements and then subsequently alter them to different groups of visitors. That way you can test what version is most attractive to what group of visitors, what group of people in terms of conversion and bounce rates.

**Chapter 28**

# Les 22: 22/05/2015

## 28.1 Slides: 7F.E-business-part3-QuizSession-StudentVersion

Heel belangrijke video voor het examen! Bv over de sessies: bounce rates.

**Cookie:**

* Small text file.
* Unique user ID: cookie hoort telkens bij een website en die cookie bevat op z’n minst een unique user ID (belangrijkste element ervan).
* Stored on hard disk.
* Privacy concerns: het feit dat die cookies lokaal worden bewaard en dat je users uniek kan identificieren kan leiden tot privacy issues.
* User consent: aangeven dat je akkoord bent met het gebruik van cookies. De gebruiker moet er van op de hoogte zijn.

**Cookie uitleg:** Hoort die altijd bij ´e´en website of niet?

First-party cookie: meest veilige cookies: cookie dat gekoppeld is aan de URL/adres van de website waar je naar surft. Stel dat bv de Standaard een cookie opslaagt omdat je op hun website zit, enkel de Standaard zal die cookie kunnen raadplegen.

Third-party cookie: worden gezet op jouw PC door een bepaalde website, maar niet door de website die je zelf bezoekt. Veel content bevindt zich op websites van adverteerders. Reclameboodschappen bv worden heel vaak gehost door verschillende domeinen. Third party kan ook een cookie op jouw systeem zetten. Is gerelateerd aan een of ander stukje content dat zich op die website bevindt. Is gevaarlijk want je kan heel veel websites aan elkaar linken en zo’n adnetworks zijn dan in staat om jou als user te tracken overheen verschillende websites. Heel veel webbrowsers bieden momenteel functionaliteit aan om die third-party cookies te blokkeren.

Verschil tussen beiden goed kennen.

**Page tagging:**

* Client side
* Data collection server: interrageren met een server.
* JavaScript
* Code snippet: in de HTML code van de website zit een JS snippet: klein stukje code dat wordt gedownload samen met de HTML code van de website die je bezoekt.
* Google Analytics: het product van Google dat werkt met page tagging en over het verzamelen van statistieken gaat.

**Page tagging uitleg:** Wat er gebeurt is: jij als user surft naar een website. Je stuurt een request naar een server om een pagina op te vragen. Die server stuurt jou de website terug: HTML-code. In die HTML code bevindt zich een snippet: een heel kort stukje code dat door het device van de user wordt ge¨ınterpreteerd. Het JS gaat ervoor zorgen dat je een bepaalde library op jouw PC zal laten werken. **Slide 10** toont een JS snippet. Google Analytics werkt met *analytics.js*-library. Eens je zo’n snippet hebt gedownload, gaat die software in jouw browser gerund worden. Zodra die snippet gelezen wordt door jouw PC, zal die *analytics.js*-library beginnen werken. De functionaliteit bevindt zich in de library die zich al op jouw PC bevindt. Die gaat data sturen naar de data collection server. Je hebt nog een library die de code effectief gaat uitvoeren. De snippet gaat enkel in staat zijn om de info te geven die de data collection server gaat contacteren. Is het omgekeerde van server logs.

**Bounce rate vs page exit rate uitleg:**

**Bounce rate:** Kijken hoeveel van jouw websitegebruikers jouw website onmiddelijk verlaten (dus niet doorsurfen naar een andere pagina).

**Page exit rate:** Interessant om te kijken naar registratiepagina’s. Als die exit rate hoog is, gaan mensen dus afhaken wanneer ze zich moeten registereren. PER kan enkel gedefini¨eerd worden voor een specifieke pagina, BR voor de site.

**Slide 13:** we hebben een aantal visits/sessies van webpagina’s. Voor de bounce rate van de site ga je kijken hoeveel keer er maar ´e´en pagina bezocht is geweest en hoe vaak dat niet is gebeurd. Die verhouding is de bounce rate. Bounce rate voor een pagina: kijken hoe vaak een pagina gebounced wordt. Moet dus al voorkomen in het bezoek. Kijk naar die pagina’s waar die pagina als eerste voorkomt. In 3/4 gevallen is B de eerste en laatste pagina die bezocht wordt. In dat ene geval (1/4), wordt pagina E ook nog bezocht.

Page exit rate: kijken hoe vaak B de laatste pagina was vs hoe vaak B in totaal bezocht is geweest. Als B meerdere keren zou voorkomen in een visit, wordt die ook meerdere keren geteld! Page exit rate zou dan bv 5/7 kunnen zijn.

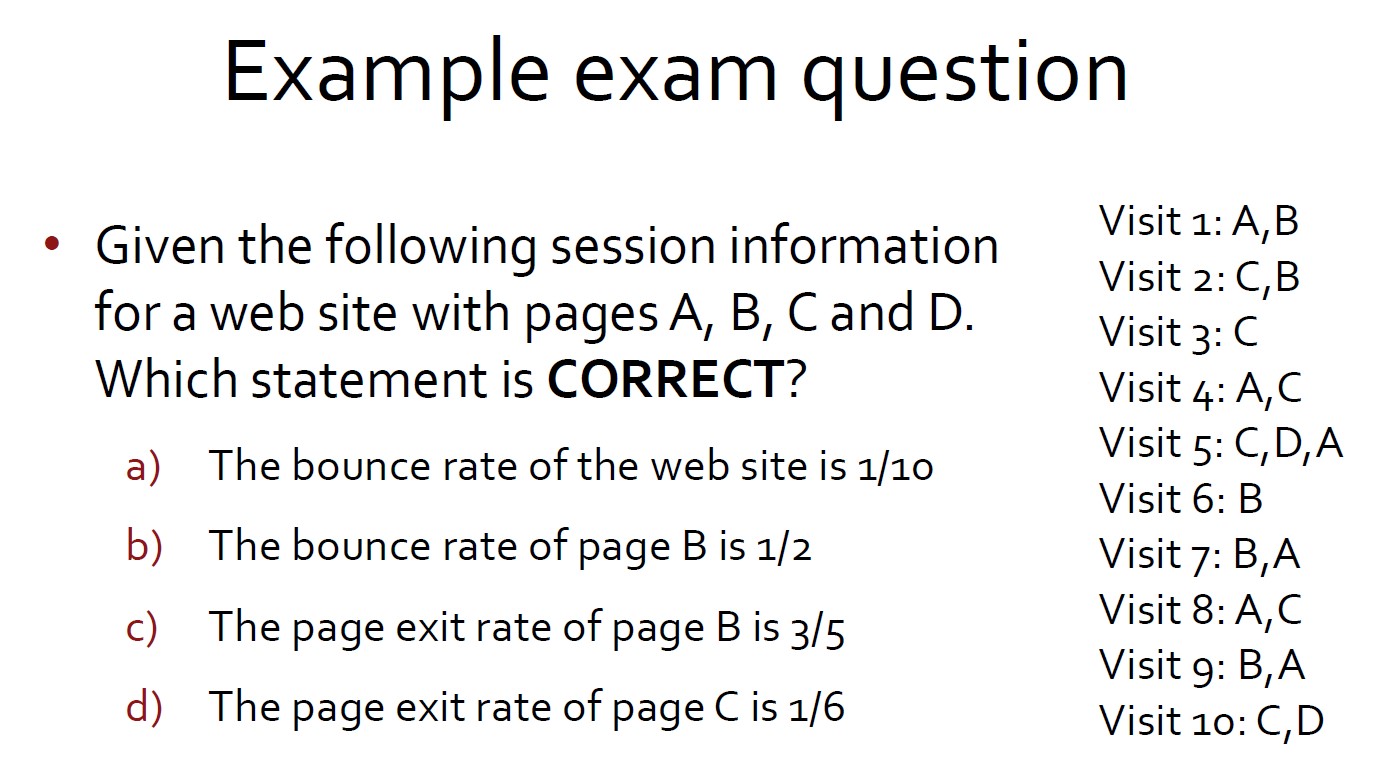


Figure 28.1: Voorbeeld examenvraag.

**Vraag hierover:** zie Figuur 28.1.

Antwoord: C

**Exit links uitleg:** Geld vragen/krijgen door te linken naar andere websites. Het is voor heel veel websites belangrijk te weten waar het uitgaande verkeer naartoe gaat vanuit hun website. Als je puur werkt op basis van server log analyse, kan je niet weten waar de gebruiker naartoe surft nadat hij jouw domein heeft verlaten. Je bent bv host van www.standaard.be en iemand klikt op een link van de vdab. Als de Standaard enkel gebruik zou maken van server analyse, zouden zij niet weten waarnaar een link wordt gegeven op de site van de Standaard. Met exit links gaat dat wel. Je kan op 2 manieren, zowel bij server log analyse als met exit links: je kan dat tracken door een external redirect. Wat er gebeurt is dat er vooraleer je de user naar de andere pagina laat surfen, je hen kort laat surfen naar een tussenpagina die niet zichtbaar is voor de gebruiker wanneer iemand op een exit link klikt. Je stuurt de gebruiker dus eerst nog naar een pagina op jouw domein. Dat kan je analyseren in een server log. Die page view gaat gelogd worden en zo kan je inzicht krijgen via een server log analyse. Je kan ook werken met JS: gebruik maken van page tagging in JS: gebruik maken van de onClick functie bv. Zodra iemand op die link klikt, gaat de JS-library een bericht sturen naar jouw server. Zowel via server log analyse als via page tagging kan je die exit links tracken. ⇒ Typische examenvraag.

**Advertising Analytics 2.0:** Komt zeer waarschijnlijk op het examen. Vooral de grotere boodschap achter het artikel.

**Serie van vragen:** zie Figuur 28.2.

**Q1:** False

**Q2:** False **Q3:** True

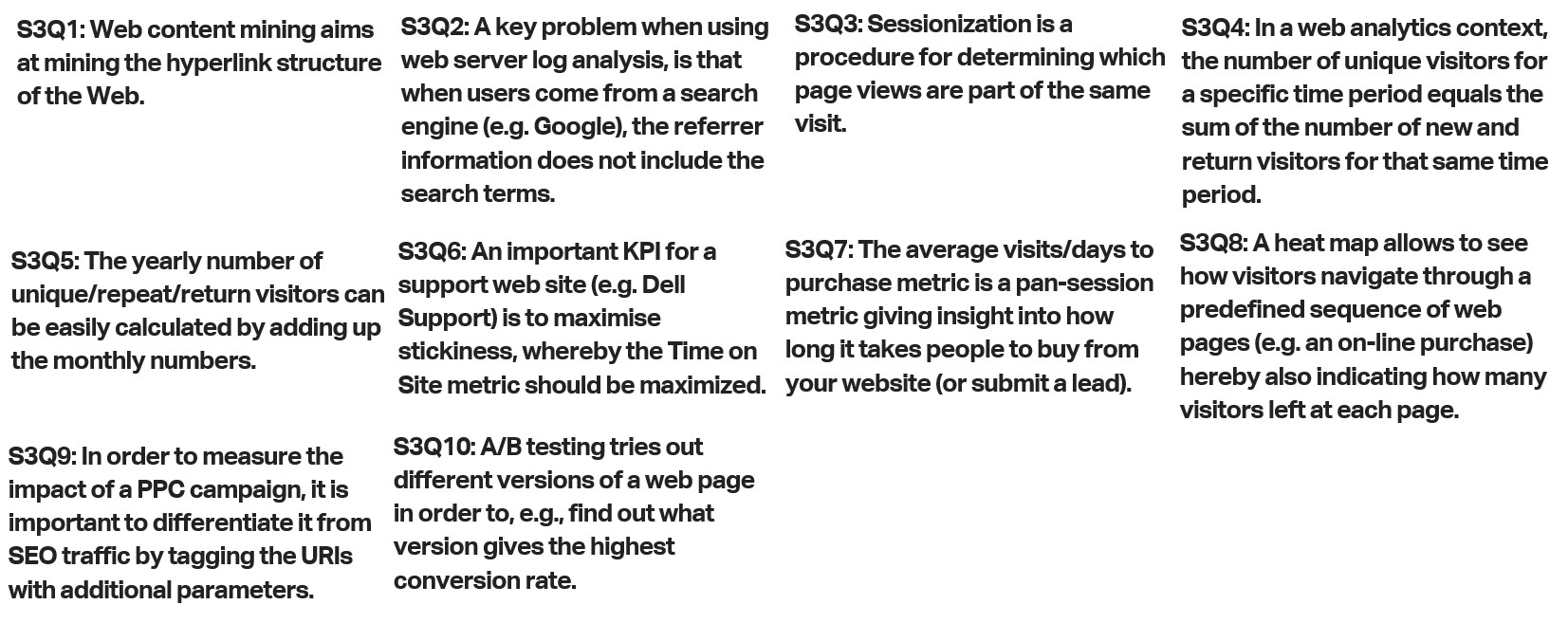


Figure 28.2: Voorbeeld examenvragen.

**Q4:** True: return vs repeat visitor. Via slides: in periode 1 zien we ABACB onze website bezoeken. Een return visitor is er een die in een vorige tijdsperiode jouw site bezocht heeft. Een repeat visitor is er een die in dezelfde periode jouw site bezocht heeft.

**Q5:** False: je kan geen unique/return visitors optellen overheen tijdsperiodes.

**Q6:** False

**Q7:** True

**Q8:** False: wat beschreven is, is een funnel plot.

**Q9:** True

**Q10:** True: belangrijk het verschil tussen A/B testing en multivariate testing te kennen! Ze lijken zeer sterk op elkaar: je designs van websites vergelijken. Bij A/B: 2 versies. Bij multivariate: kijken naar verschillende elementen gelijktijdig. Je gaat meer dan 3 of 4 versies controleren. Je gaat meerdere zaken testen. Bij A/B gewoon pagina per pagina.

## 28.2 Slides: WrapUp

**Examen:** Gesloten boek, multiple choice, ongeveer 15 vragen van elke prof (dus een 30 in totaal). Met giscorrectie. Correct: +1, niks: 0, fout: -1/3. Volg een goede strategie: als 5 mogelijkheden en over 3 twijfel je, heb je 1/3 om juist te antwoorden. Als je dit doet bij 3 vragen, heb je in principe nog steeds 0.333/3, wat beter is dan 0/3. Kans is op zich klein dat je -1/3 haalt. Er wordt getest of je de vragen begrijpt. Je moet tonen dat je inzicht hebt in wat er verteld is. Begrijpen van de dingen + 6 a 7 vragen van Prof. De Weerdt zijn oefeningen.

**Voorbeeldvragen: Slide 13:** Oefening.

**Slide 14:** Antwoord: B: heeft niet te maken met toetredingsbelemmeringen (of de vermindering ervan). Internet wel.

**Slide 15:** D: Dice en geen slice. Dice gaat kijken naar meerdere dimensies, slice gaat kijken naar 1 dimensie. **Slide 16:** C is fout: feitentabel bevat de samengestelde sleutel die verwijst naar de dimensietabellen.

1. We weten dus dat er 1000 man sowieso gaat reageren, in het perfect model zouden we dus door 10% van de 10 000 aan te spreken, al die 1000 mensen getarget hebben (vandaar de 100%). Indien we no model zouden gebruiken, zou slechts 10% van die 1000 mensen die sowieso zouden reageren getarget zijn. Door het response model te gebruiken, verhogen we dat percentage. Bij het aanschrijven van 30% van het klantenbestand, zal dus 65% van de 1000 mensen die sowieso gingen reageren getarget zijn. [↑](#footnote-ref-1)