Hoofdstuk 2 : Data Flow Diagrammen

Data Flow Diagrammen (DFD), 'datastroomdiagrammen' in het Nederlands, geven een functiegerichte visie van de transformaties van invoer naar uitvoer.

2.1 Inleiding

Bij verschillende bedrijven vind je systemen terug die taken automatiseren. Om taken te kunnen automatiseren in een systeem, heb je data nodig als input en zul je ook data weergeven als output. DFD's zijn ideaal om deze scenario's te visualiseren. DFD's helpen om een overzicht te krijgen van de gegevensbronnen, de gegevensstroom en de bedrijfsprocessen die deze gegevens verwerken. Een DFD toont welke informatie nodig is binnen een proces, waar deze opgeslagen is en hoe deze informatie doorheen het systeem stroomt om een bepaald doel te bereiken. Zoals de naam Data Flow Diagram al laat uitschijnen, laat deze de stroom van gegevens zien in een systeem.

Vergis je wel niet met flowcharts. Bij deze laatste speelt in tegenstelling tot DFD's, oorzaak-gevolg en de sequentie van processen ook een belang.

Daarom worden DFD's ook aanzien als gebruiksvriendelijk, zowel voor de ontwerper als de eindgebruiker om te interpreteren. DFD's worden daarom zowel gebruikt in de verkennende als ontwikkelingsfase van een project. Hoewel data flow diagrammen zeer gebruikelijk zijn bij verschillende organisaties, sommige analisten kennen ze onder een andere naam. Zoals Yourdon, de pioneer van software engineering methodes, ooit heeft vermeld in een artikel over structured analysis, refereren sommige organisaties naar data flows diagrams als 'Bubble chart', 'Bubble diagram', 'Process model' (of 'business process model'), 'Business flow model', 'Work flow diagram' of 'Function model'.

Data flow diagrammen worden meestal ingedeeld in verschillende numerieke niveaus om meerdere granulariteiten van een bedrijfsproces weer te geven, met niveau 0 – ook wel bekend als een context diagram of figuur 0 – zijn hoogste niveau. Niveau 1 is het op één na hoogste niveau van de informatiestroom, 2 is meer korrelig dan 1, 3 is meer gedetailleerd dan 2, en zo verder (hoewel weinig systemen meer dan 3 niveaus weergeven). Deze methode is vooral handig bij complexe bedrijfsprocessen, omdat het een business analist in staat stelt om een het hele bedrijfsproces bondig te illustreren, en toch zo gedetailleerd als nodig is. Deze techniek maakt dus gebruik van een top-down methode, om op een gerichte manier een uitbreiding van de analyse uit te voeren. Daarom, als een gebruiker vele processen en niveaus van data heeft om in een diagram (deelprocessen) op te nemen, worden best meerdere niveaus gebruikt.

Voorbeeld van het Processymbool:

Figuur 2.1

Voorbeeld van het data store symbool:

Figuur 2.2

2.2 Welke elementen heeft een DFD?

Een DFD omvat de gegevens, processen, dataopslag en externe entiteiten van een systeem en alle gegevens die nodig zijn voor het systeem om te functioneren (zowel hoe de data stroomt en waar de data wordt opgeslagen). Daartoe omvat het meestal notaties (of symbolen) van één van de twee tradities: Yourdon & Coad of Gene & Sarson. Deze tradities zijn slechts twee verschillende stijlen van symbolen, maar beide tonen dezelfde dingen: processen, dataopslag van gegevensstromen, en externe entiteiten. De stijl van de symbolen die een analist volgt, is niet zo belangrijk. Het is wel belangrijk dat een analist consequent omgaat met de symbolen. Wij zullen de Gene & Sarson symbolen gebruiken. Een kort overzicht van de elementen een data-flowdiagram's volgt hier onder.

2.2.1 Proces

Een proces wordt ook vaak aangeduid als een 'bubble, functie of een transformatie'. Zijn functie is de inkomende datastroom om te vormen tot een uitgaande datastroom. Met andere woorden, een proces geeft weer hoe informatie beweegt in het systeem. Een proces moet worden benoemd op basis van wat het precies doet (bijvoorbeeld: 'Verwerk orders', 'Registreer klant' of 'Geef verkoopcijfers van verkoper'), en zou in- en uitgangen (gegevensstromen, hieronder beschreven) moeten hebben. Processen zonder in- of uit-informatiestromen worden een 'oneindige put' en zijn logisch inconsistent.

Zoals je in Figuur 2.1 kan zien is het symbool voor een proces, volgens Gene & Sarson notatie, een rechthoek met afgeronde hoeken.

2.2.2 Data store

Een data store of dataopslag in het Nederlands, is een opslagplaats voor gegevens die worden gebruikt binnen het systeem.

Een data store kan variëren van een database, een tekstfile, een internetbron,… Als naamgeving voor een data store gebruiken we een naam die duidelijk weergeeft wat er in de data store zit en is dus ook altijd in het meervoud, bijvoorbeeld: Klanten, Studenten, Producten, Adressen. Een data store wordt gesymboliseerd door een rechthoek zonder rechter kantlijn zoals weergegeven in Figuur 2.2.

2.2.3 External entities

External entities of externe entitieten zijn externe bronnen en bestemmingen van informatie die relevant zijn voor het systeem. Voorbeelden van externe entiteiten kunnen klanten, leveranciers, of externe databases zijn. Een externe entiteit wordt voorgesteld door een rechthoek.

2.3.4 Data flow

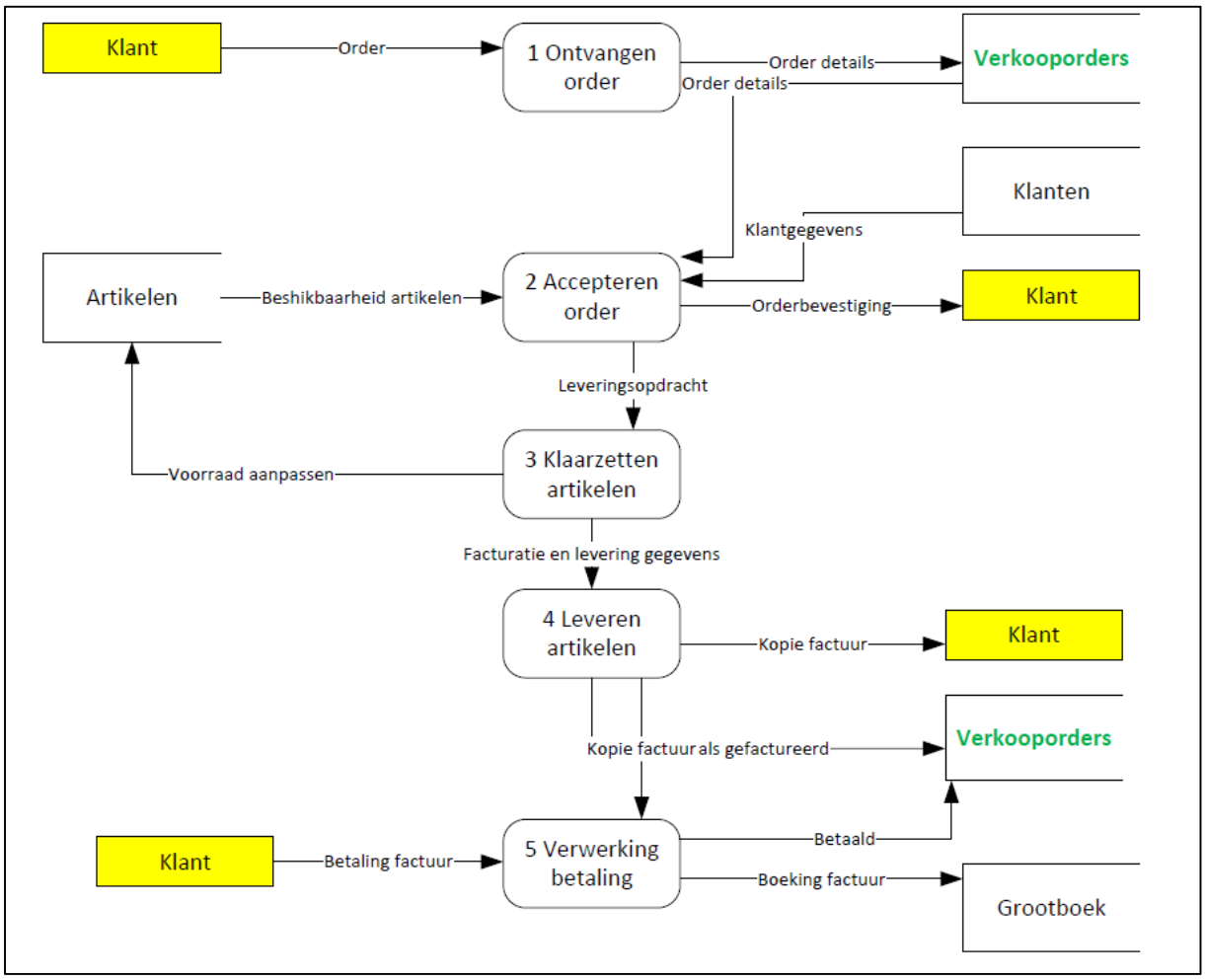
Data flows of gegevensstromen in het Nederlands, laten zien hoe informatie beweegt binnen een systeem. Ze zijn de 'pijpleidingen' waardoor de data stroomt. De stromen vertegenwoordigen 'gegevens in beweging', terwijl de datastores 'gegevens in rust' vertegenwoordigen. Gegevensstromen worden gepresenteerd als pijlen en gelabeld volgens de gegevens die ze vertegenwoordigen (bijvoorbeeld: 'Aankooprecords', 'Geupdate klantgegevens' of 'Dagelijkse aankooptrends').
Je kan niet willekeurig pijlen tekenen, zo kan er geen pijl zijn:

- Tussen een data store en een andere data store: dit zou betekenen dat een data store zelfstandig zou kunnen beslissen om data naar een data store te verzenden, dit kan niet en moet dus via een proces gaan.

- Tussen een data store en een externe entiteit: dat zou enkel kunnen als de externe entiteit kan lezen van en schrijven naar een data store, dit mag niet en moet via een proces verlopen.

Meestal wordt ook vermeden om een data flow te tonen tussen twee externe entiteiten. Bijvoorbeeld een klant geeft info aan verkoper en die laatste werkt met systeem. De communicatie tussen klant en verkoper gebeurt niet in het systeem en we willen met DFD's vooral weergeven wat er in het systeem gebeurt.

2.3 Tips om DFD's te tekenen



Figuur 2.3

In Figuur 2.3 vind je een DFD van een verkoopproces. Merk op dat de externe entiteit Klant meerdere keren voorkomt. Dit gaat natuurlijk telkens om dezelfde externe entiteit, maar indien we maar één keer Klant in DFD zouden tekenen zou er een wirwar aan pijlen ontstaan. Om duidelijk identieke objecten aan te geven kan er met kleuren gewerkt worden. Zo komt ook Verkooporders twee maal voor. Het werken met kleuren of andere symbolen is niet verplicht en mag zeker geen omgekeerd effect hebben, dus dat er onduidelijkheid ontstaat.

Verder neem je ook best volgende punten in rekening:

- Alle data flows worden gelabeld en beschrijven de informatie die wordt door gegeven.
- Meestal is de diagram beter leesbaar als de processen in het midden staan, de externe entiteiten links en data store aan de rechterkant van het diagram.
- Begin het proces met een sterk werkwoord aangevuld met een object, geef vooral de functie weer, niet de naam van de uitvoerder.
- Elk proces moet een ingang en een uitgang hebben. Als er geen output is heeft een proces geen zin. Zonder input, kan er ook geen output gegenereerd worden

• Elke data store moet ten minste een inkomende gegevensstroom hebben en een uitgaande datastroom. Als gegevens niet worden uit gelezen is het de vraag of deze moet worden opgeslagen. Bovendien moeten in de eerste plaats er gegevens in de data store zitten dus is het onwaarschijnlijk dat er geen schrijfopdracht is naar de data store.

• Data flows kunnen gaan van

  – Externe entiteit naar proces en vice versa

  – Proces naar proces

  – Proces naar data store en vice versa

• De sequentie van de nummering van processen is willekeurig. In Figuur 2.3 kan er wel een logische volgorde gezien worden in de nummering maar dat is als het ware toeval.

• Vermijd al the complexe DFD's.

• Teken een DFD zo vaak als nodig opnieuw, dus heb geen schrik voor wijzigingen. Maak dit ook altijd duidelijk aan de klant, want computerdiagrammen lijken voor een klant vaak definitief.

Tekenen van diagrammen als dit vereist oefening. Je moet je niet onnodig zorgen maken als jouw diagram er niet precies hetzelfde uit ziet als van een collega of een model antwoord. Een gegevensanalyse wordt zeer zelden gedaan in isolatie en u en uw collega's en de eindgebruiker zullen tot een consensus komen over het finale model.

2.4 Ontwikkeling van (gelaagde) DFD's

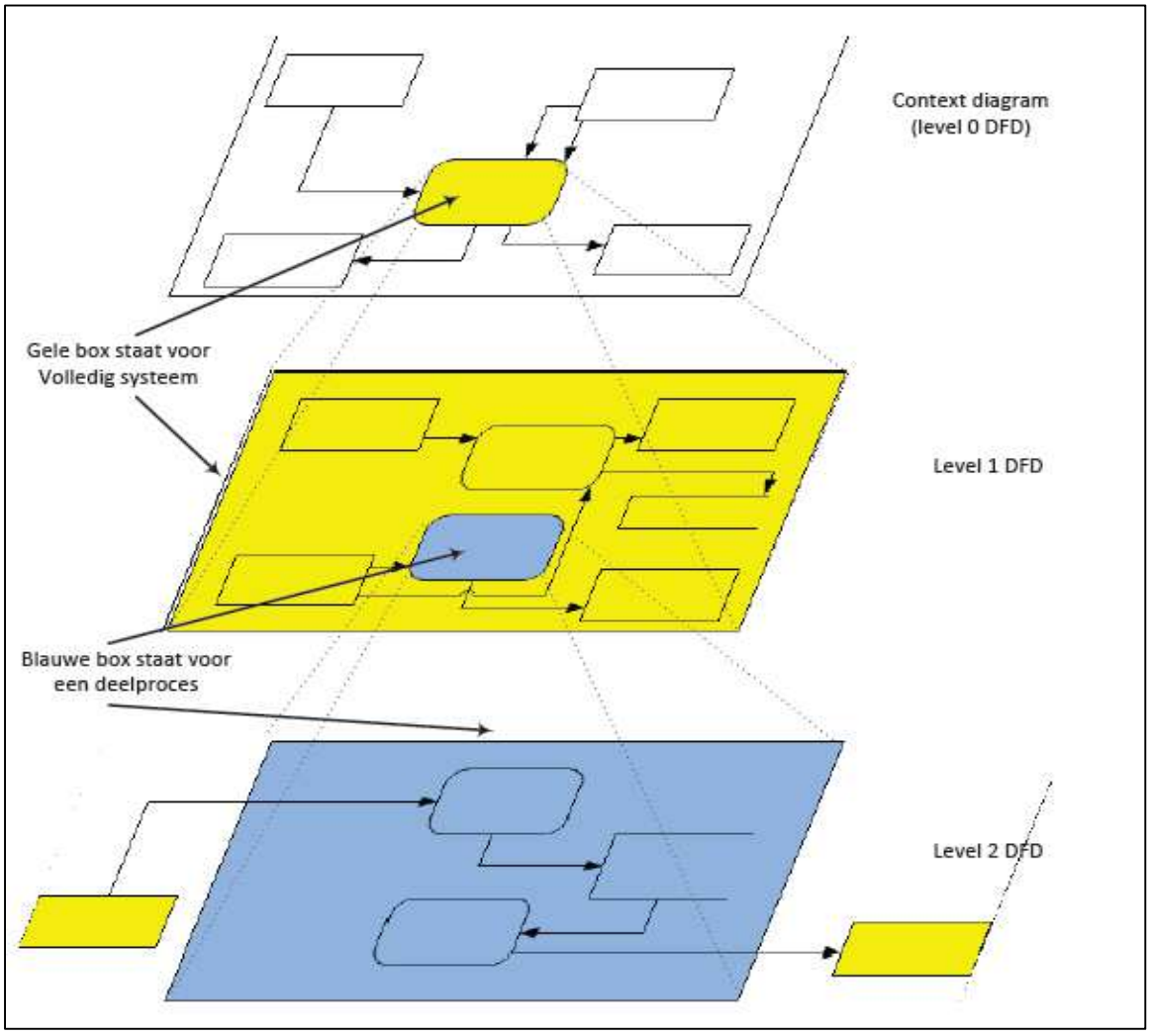
Data flow diagrammen komen meestal in sets. Een set bestaat uit verschillende niveaus. We beginnen met een contextdiagram. Dit toont de mensen en/of systemen (=externe entiteiten) die interageren met het systeem in ontwikkeling. Met interactie bedoelen we plaatsen van informatie in, of het nemen van informatie uit ons systeem. Dit contextdiagram geeft een overzicht van de informatie die binnenkomt in en buitengaat uit het systeem. Het systeem wordt vertegenwoordigd door een box. In deze DFD (niveau 0 DFD) worden geen processen of data store weergegeven.

Een Ander diagram (niveau 1 DFD) wordt daarna ontwikkeld die de processen toont, die nodig zijn om de inkomende informatie (input) in het context diagram om te zetten naar de uitgaande informatie (output). In deze DFD wordt in detail weer gegeven welke processen verantwoordelijk zijn voor het accepteren van de verschillende input en de productie van de verschillende output. Als deze processen nog meer uitwerking vergen wordt dit getoond in een ander diagram (niveau 2 DFD) waar deelprocessen worden weergegeven tezamen met de nodige extra data stores. Figuur 2.4 geeft weer hoe deze verschillende niveaus met elkaar gelinkt zijn.

2.4.1 Contextdiagram

Het contextdiagram, of niveau/level 0 DFD, geeft een overzicht van de input en output van het systeem. Het toont ook de externe entiteiten die data verstrekken of ontvangen. Deze komen meestal overeen met de mensen die gebruik maken van het systeem dat we ontwikkelen.

Het context diagram helpt om onze systeemgrens te definiëren, om zo te laten zien wat er is opgenomen in, en wat is uitgesloten van, ons systeem. Het schema bestaat uit een rechthoek die de systeemgrens, de externe entiteiten interactie met het systeem en de gegevens die stromen in en uit het systeem. In Figuur 2.4 hieronder vind je een voorbeeld van een contextdiagram. Nu rond contextdiagrammen vind je ook verschillende interpretaties. Sommige geven in een contextdiagram enkel het centrale systeem, de externe entiteiten en de pijlen (zonder uitleg).
Wat wij in deze cursus als contextdiagram beschouwen, wordt door anderen als Level 0 gezien eventueel met meerdere processen maar nog zonder data stores. Hoe je het ook aanpakt, het contextdiagram moet vooral weergeven wie de externe entiteiten zijn en mag zeker geen data stores bevatten. Verder levels van DFD moeten telkens meer details geven waarbij consistentie tussen de verschillende niveaus belangrijk is.

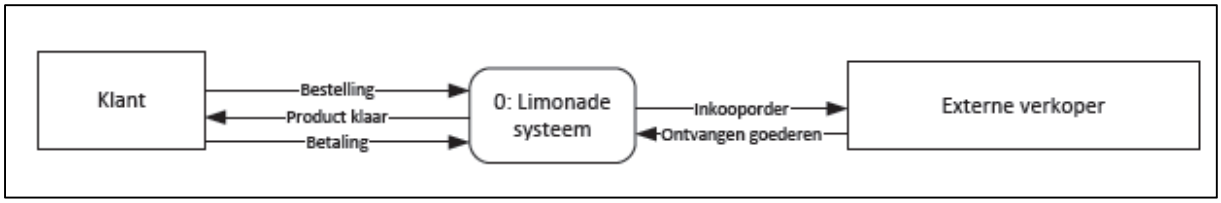


Figuur 2.4: Verschillende niveaus van een DFD

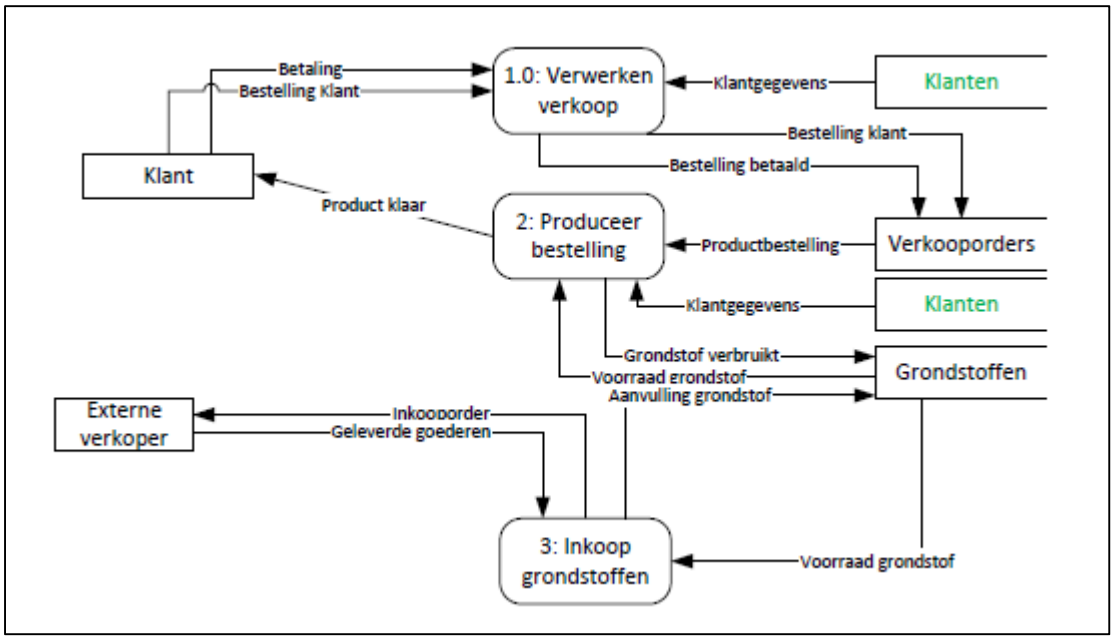
2.4.2 Level 1 DFD

Nu willen we in een model verder uitwerken wat het systeem zal doen met de informatie die de externe entiteiten aanleveren. We maken een diagram dat, in feite, een vergrootglas is om de systeemgrens van het proces in het contextdiagram meer in detail te bekijken. We zullen kijken binnen de rechthoek en weergeven wat gebeurt met de input om de vereiste output te verkrijgen.

Het niveau 1 DFD dat we bouwen is een “kind schema” van het contextdiagram. We moeten alle ingangen, uitgangen en externe entiteiten die aanwezig zijn in het context diagram terug zien. Vaak is men bij het maken van de eerste level 1 DFD, onzeker over hoe veel processen er nu getoond moeten worden; Dit hangt enigszins af van de case study waar we verder op terugkomen. Echter, als richtlijn kan je nemen, dat je niet meer dan acht of negen processen zou mogen tonen. Meer dan dit zou het diagram te rommelig maken. Daarnaast mag je aannemen dat er meestal ten minste drie processen zijn. Minder dan zou dit betekenen dat het systeem onrealistisch eenvoudig is.



Figuur 2.5: Voorbeeld van een contextdiagram



Figuur 2.6: Voorbeeld van een level 1 DFD op basis van het contextdiagram in Figuur 2.5.

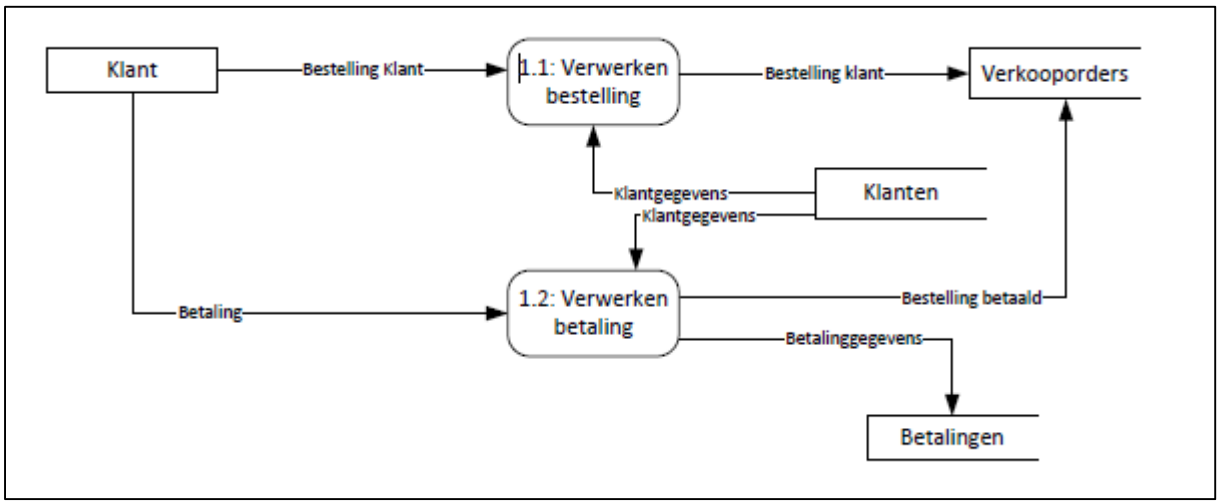
Figuur 2.6 toont een level 1 DFD van het contextdiagram in Figuur 2.5. Merk op dat hier de nummering begint met 1.0, terwijl de andere processen 2 en 3 zijn, en dat de nummering in het contextdiagram 0 was. Nu om duidelijk aan te geven dat een proces nog verder uitgewerkt wordt schrijven we 1.0 i.p.v. 1.

2.4.3 Verdere levels van DFD's

Het proces "Verwerken verkoop" uit Figuur 3.6 zullen we in meer detail uitwerken. In dit geval maken we dus een Level 2 DFD getoond in Figuur 3.7.

Aangezien dit deelprocessen van proces nummer 1.0 uit level 1, nummeren we ook verder met 1.1, 1.2, enzovoort. Moest proces 1.1 nog verder uitgewerkt worden zou dit nummer 1.1.0 krijgen. Hou bij lagere niveaus van DFD's rekening met:

- Dat alle data flows die in het parent-proces staan ook in deze DFD voorkomen.
- Dat je aan de nummering goed kan zien welk proces sub-processen heeft en bij wel proces, een sub-proces hoort.
- Dat niet alle systeemprocessen evenveel levels hebben.



Figuur 2.7: Voorbeeld van een level 2 DFD op basis van "Verwerken verkoop" uit de level 1 DFD in Figuur 2.6.

Het kan altijd zijn dat je op lagere niveaus nog nieuwe data store introduceert, maar er kunnen t.o.v. het contextdiagram geen externe entiteiten bijkomen.

2.4.4 Hoe een DFD in lagen ontwikkelen?

De top-down voorstelling is het eindproduct en bevordert de leesbaarheid. Gelaagde DFD's worden niet noodzakelijk ook top-down ontwikkeld, integendeel. De context-diagram is er meestal vrij snel en is meestal het eerste DFD, maar daarna gaat men meestal deelprocessen visualiseren en pas daarna wordt gekeken hoe alles samenhangt.

Beste ga je als volgt te werk:

- Som alle basisactiviteiten op, in korte zinnen, die in het systeem (moeten) plaats vinden.
- Denk ook na over aanvullende activiteiten die nodig zijn om de basisactiviteiten uit te voeren.
- Groepeer deze activiteiten in logische groepen. Deze groepen zullen de systemen van level 2 vormen.
- Tot slot kan je DFD level 1 maken.