WORDT
NIET UITGELEEND

Afstudeerverslag

Pieter Visser

Rijksuniversiteit Groningen
&
KeyMark Services

18 juli 2001

Pieter Visser
Soephuisstraatje 3a
9712 BZ Groningen

Inhoudsopgave

1 INLEIDING 3

1.1 ENIGE TERMINOLOGIE 3

2 HET BEDRIJF 4

2.1 KEYMARK SERVICES 4
2.1.1 De bedrijfsstructuur 4
2.1.2 Productieomgevingen Deventer en Groningen 5
2.1.3 Mijn aandeel in de zaak 5
2.1.4 De controle 6
2.2 STROOMSCHEMA 6
2.3 INFORMATIESTROMEN EN GANG VAN ZAKEN 7
2.3.1 Gang van zaken in het algemeen 7
2.3.2 Gang van zaken op zaterdag 7
2.3.3 Communicatie (informatiestromen) 8
2.4 DE TOEKOMST 8

3 DE OPDRACHT 9

3.1 PROBLEMEN 9
3.2 GEWENSTE SITUATIE 9
3.3 DE OPDRACHT 9

4 HET ONTWERP VAN DE DATABASE 11

4.1 HET DATAMODEL 11
4.1.1 Dataflow 11
4.1.2 Gevolgen voor de database 12
4.1.3 Relationeel model 13
4.2 DE EERSTE OPZET 14
4.2.1 De eerste implementatie 15
4.2.2 De tweede poging 15
4.3 UITEINDELIJKE DATABASE 16
4.3.1 Tabelindeling en inhoud 16
4.3.2 Views en hun inhoud 20
4.3.3 Datadiagram 21
4.4 HISTORISCHE GEGEVENS IN DE DATABASE 21
4.5 OPMERKINGEN OVER DE IMPLEMENTATIE 22
4.6 BEVEILIGING 23

5 GEBRUIKTE ALGORITMES 24

5.1 ALGORITME VOOR PLANNING 24
5.1.1 Handmatige versie 24
5.1.2 Geïmplementeerde versie 24
5.2 ALGORITME VOOR UITDELING 26
5.3 ALGORITME VOOR CONTROLE 26
5.3.1 Handmatige versie 26
5.3.2 Geïmplementeerde versie 26
5.4 OVERIGE ALGORITMES 28
5.4.1 Inlezen van vacaturebestanden 28
5.4.2 Bepalen van weeknummers 28

6 HET ONTWERP VAN DE APPLICATIE 30

6.1 FUNCTIONEEL ONTWERP 30
6.1.1 Doel ontwerp 30
6.1.2 Algemene functionaliteit 30
6.1.3 Menustructuur 31
6.1.4 Onderdelen van het menu 31
6.1.5 Opmerkingen 35

6.2 IMPLEMENTATIETRAJECT 35
6.2.1 Doel van de opdracht 35
6.2.2 Algemene technieken 35
6.2.3 Planning 37
6.2.4 Uitdeling 39
6.2.5 Controle 39
6.2.6 Onderhoud 40
7 VERDERE ONTWIKKELMOGELIJKHEDEN 42
7.1 AANBEVELINGEN 42
7.2 OVERIGE OPTIES 43
8 CONCLUSIE 44

1 Inleiding

Dit afstudeerverslag is het resultaat van ongeveer 11 maanden werk aan een afstudeeropdracht bij Keymark Services. In dit verslag wordt achtereenvolgens de organisatie, de opdracht en de uitwerking van die opdracht beschreven. In de conclusie aan het einde worden mijn ervaringen beschreven.
De documentatie bij de opdracht bestaat uit drie delen: dit verslag, een gebruikershandleiding en een technische handleiding. Dit verslag gaat voornamelijk over de door mij bewandelde weg, hoe de opdracht is uitgewerkt en wat ik daarvoor heb moeten doen. De gebruikershandleiding is van belang als men met de gemaakte applicatie wil werken. De technische handleiding tenslotte is noodzakelijk voor het begrijpen van de achterliggende techniek.

1.1 Enige terminologie

In de processen binnen het bedrijf worden een aantal termen gebruikt die enige uitleg nodig hebben. Hieronder wordt een aantal woorden uitgelegd die in de rest van dit verslag voor zullen komen.

BBS	-	Bulletin Board System, mag gezien worden als de voorloper van Internet. Krijgt via een modem een rechtstreekse verbinding met de gebruiker, wordt gebruikt om bestanden over te brengen.
Bron	-	Naam van de krant of website waarin (waarop) een vacature staat.
Controleur	-	Medewerker die het proces begeleidt, vacatures uitdeelt, terug ontvangt en de inhoud controleert.
Vacature	-	Personeelsadvertentie in een bron.

Verder is het van belang te weten dat de binnen het bedrijf de weeknummering (het nummer van de week in een bepaald jaar) soms afwijkt van de officiële weeknummering. Bij alle contacten met medewerkers wordt gewoon gebruik gemaakt van het nummer van de week zoals dat op de kalender staat. Wanneer wordt gesproken over de database, wordt er met het weeknummer altijd het nummer bedoeld van deze database. De database wordt elke maandag bij klanten bezorgd, en hierbij is het weeknummer dus met één verhoogd.

Met een *venster* (Engels: *window*) wordt een venster binnen het programma bedoeld, zo'n scherm is dus fysiek binnen het hoofdscherm van de applicatie te vinden. Met een *scherm* (Engels: *window*) wordt een scherm bedoeld dat ook "voor" de applicatie kan staan. Vaak is het ook nodig zo'n scherm af te sluiten voordat de applicatie zelf doorgaat.

2 Het Bedrijf

In dit hoofdstuk wordt de opdrachtgever beschreven. Vervolgens komt de geschiedenis van het bedrijf kort aan bod. Verder gaat het over mijn relatie met het bedrijf en hoe deze tot stand is gekomen.

2.1 KeyMark Services

KeyMark Services v.o.f. is in 1997 ontstaan uit een joint venture van VNU Business Publications en Pelican Services.

Het bedrijf richt zich op de arbeidsmarkt en verzamelt personeelsadvertenties (vacatures) uit (gedrukte) media. Deze vacatures worden samengevat en in verkorte vorm in een database opgeslagen. De gegevens worden verwerkt en er wordt extra informatie aan toegevoegd, de zogeheten verrijking. Daarna worden de vacatures toegevoegd aan de zogeheten ontwikkeldatabase. Deze bevat alle vacatures van de afgelopen jaren.
Tenslotte worden de vacatures (van de laatste 2 of 3 weken) en/of statistieken daarvan verkocht aan klanten, onder andere in de vorm van een vacaturebank en op internet.

2.1.1 De bedrijfsstructuur

Het bedrijf is op dit moment verdeeld over 3 plaatsen. Het hoofdkantoor in Amsterdam, de primaire productieomgeving in Groningen en de secundaire productieomgeving in Deventer. De verrijking en distributie van de bestanden vinden op dit moment nog plaats in Sassenheim, maar deze worden op korte termijn naar Groningen verplaatst.

In Groningen wordt het proces van vacatures verzamelen en verdelen over de werknemers gecoördineerd. Door de week worden vacatures in vakbladen en weekbladen geteld en meegegeven aan of per post gestuurd naar thuiswerkers. Op zaterdag gaat het voornamelijk om de landelijke dagbladen. Voor de regionale (dag)bladen is er een ander systeem. Er worden afspraken gemaakt met thuiswerkers die deze bladen in hun omgeving kunnen krijgen. Zij tellen de vacatures zelf en voeren ze daarna in.

Voor het coördineren zijn twee vaste fulltime medewerkers door de week en drie tot vijf parttime medewerkers in het weekend nodig. Verder wordt er gebruik gemaakt van 60 tot 80 oproepkrachten (thuiswerkers) die verspreid zijn over het hele land. Deze thuiswerkers worden op stuksbasis betaald, hetgeen inhoudt dat ze zelf moeten doorgeven hoeveel ze gedaan hebben.
De stuksprijs is afhankelijk van de grootte van een vacature. Dit is te vergelijken met de hoeveelheid werk en tijd die het kost om een vacature samen te vatten. De thuiswerkers moeten zelf bijhouden hoeveel van de vacatures groot, middelgroot of klein zijn. Dit wordt de *verdeling* genoemd. Uiteraard wordt steekproefsgewijs gekeken of deze verdeling niet te positief uitvalt voor de medewerkers.

De vaste medewerkers verdelen het werk over de oproepkrachten. Aangezien zaterdags de meeste vacatures in de landelijke en regionale dagbladen staan, wordt het meeste werk op zaterdag gedaan. Van tevoren wordt al een lijst gemaakt wie die week beschikbaar zijn. Er wordt een voorlopige indeling gemaakt wie wat gaat doen. 's Morgens wordt alles pas echt verdeeld. Alle vacatures worden gedurende de dag ingevoerd in een speciaal invoerprogramma en daarna (vanuit het land) opgestuurd via modems naar het BBS in

Groningen. Het invoerprogramma is op dit moment nog een snelle DOS applicatie, maar zal in de loop van dit jaar vervangen worden door een Windows versie.

De gehele zaterdag wordt alles wat wordt opgestuurd gecontroleerd. De zaterdagavond is gereserveerd voor de eindcontrole. Aan het eind van de avond wordt het gecontroleerde bestand opgestuurd naar Sassenheim waar de nabewerking (automatisch) plaatsvindt.

2.1.2 Productieomgevingen Deventer en Groningen

In Deventer is onlangs een nieuwe vestiging geopend waar zaterdags zes medewerkers zitten, waaronder één van de parttimers uit Groningen. Deze locatie is voornamelijk gekozen omdat zij meer in het midden van het land ligt. Van hieruit kunnen gemakkelijk meerdere dagbladen in korte tijd gehaald worden. Op dit moment wordt bijvoorbeeld het Utrechts Nieuwsblad 's morgens in Utrecht gekocht en naar Deventer gebracht. Op deze manier is deze krant al om 10:30 uur aanwezig in Deventer. In Groningen komt de krant op zijn vroegst om 11:30 aan.

De vestiging in Groningen is van oorsprong de productievestiging geweest voor **De Sollicitatiekrant**, een krant met vacatures voor middel en hoger opgeleiden. Deze krant werd uitgegeven door **IntraMedium**, opgericht door Marco Keupink in 1995. Het proces van vacatures verzamelen is overgenomen door KeyMark Services. Hier wordt nu dus de grondstof geleverd voor de rest van het proces.

2.1.3 Mijn aandeel in de zaak

Ik ben zelf in 1996 begonnen te werken voor De Sollicitatiekrant. Het werk bestond toen alleen uit het samenvatten van vacatures. Deze verkorte versies werden in De Sollicitatiekrant gezet. De krant bevatte toen ongeveer 2000 (later 3000) vacatures. Later heb ik meegewerkt aan de voorbereiding van het drukken en distribueren van de krant.

De krant verschijnt inmiddels al lang niet meer. Het aantal vacatures dat per week verwerkt wordt is nu meer dan 9000 (waarvan meer dan de helft voor middel en hoger opgeleiden). Dit aantal zou niet eens meer in zijn geheel in een krant passen.

Omdat door de toename van vacatures de hoeveelheid controlewerk te veel begon te worden voor één persoon, is besloten een controlegroep te vormen. Hiervoor zijn de beste medewerkers in Groningen gekozen die regelmatig op zaterdag aanwezig waren. Zij werkten vanaf dat moment op kantoor en bleven na hun samenvatwerk om 's avonds het werk van de andere medewerkers te controleren.
In het begin bestond de groep (inclusief mijn opdrachtgever en ikzelf) uit ongeveer 7 personen. Een aantal van deze medewerkers zijn na afronding van hun studie of wegens verhuizing gestopt. Hiermee vielen ook die personen weg die de kennis van de programma's en computers hadden. Deze rol is toen onmiddellijk overgedragen aan mij. Zo heb ik de officieuze titel "Automatiseerder" gekregen.
Op dit moment ben ik, op mijn opdrachtgever na, degene die er het langste werkt. Ik ben dan ook indirect bekend met de meeste processen. De controlegroep bestaat nu (weer) uit 5 personen.

2.1.4 De controle

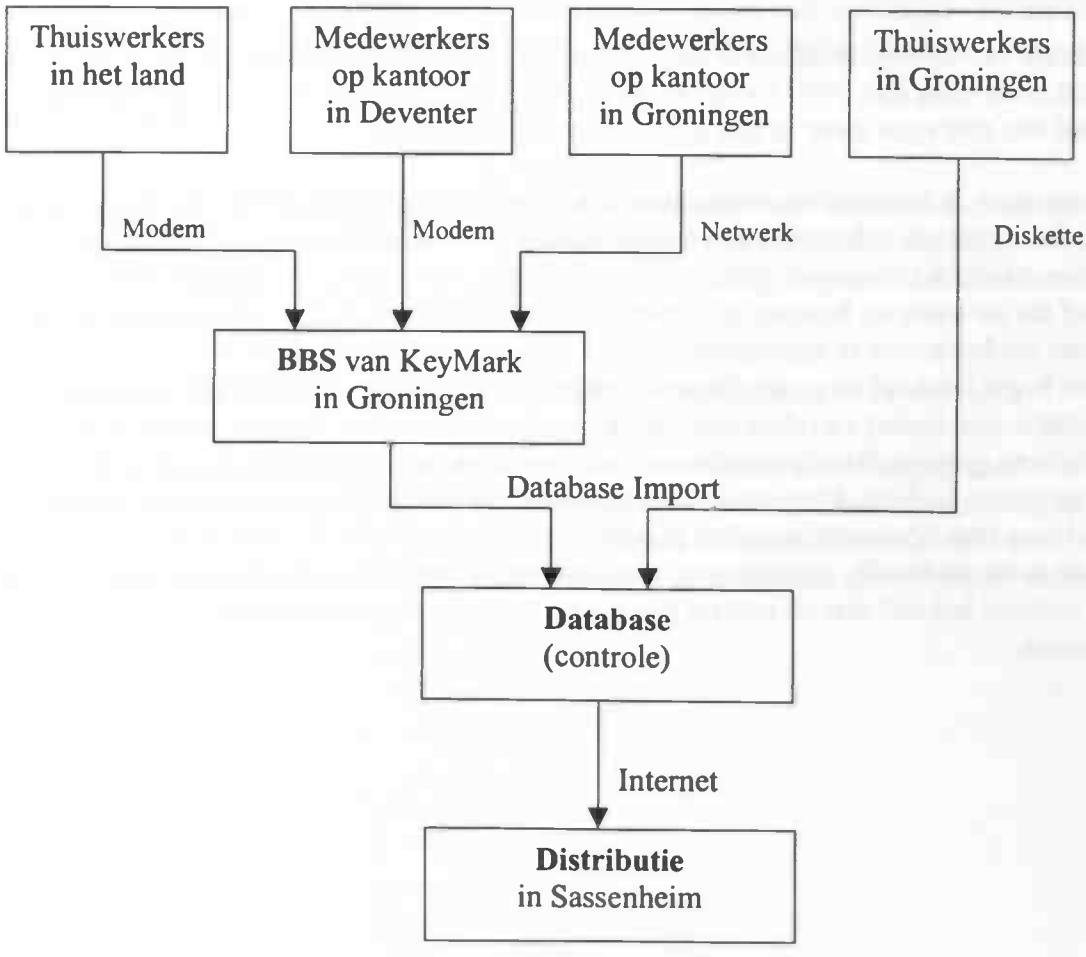
Onder controle in Groningen worden de volgende dingen verstaan:

- Herstellen van gebruikelijke typefouten, fouten in het samenvatten, hanteren van verkeerde principes, gebruiken van verkeerde afkortingen van de medewerkers.
- Vergelijken van afgesproken hoeveelheid werk en de werkelijk ingeleverde hoeveelheid. Hier mag niet al te veel verschil in zitten. (*)
- Kijken of het juiste werk gedaan is en is opgestuurd (het komt vaak voor dat medewerkers niet opletten bij het verwerken of versturen). (*)
- Bijhouden hoeveel de medewerkers gedaan hebben, hoeveel ze verdiend hebben en hoe vaak ze iets opgestuurd hebben. (*)
- Capaciteit opnieuw verdelen als er gedurende de dag blijkt dat er te veel werk is voor een bepaalde regio, of als er iemand ziek wordt. (*)

De met een (*) gemarkeerde punten moeten voor het grootste deel worden geautomatiseerd. De overige controle omvat een groot deel van het proces, maar is een onderdeel dat de uiteindelijke kwaliteit (van de opgeleverde database) waarborgt. Deze heeft echter vrijwel niets meer te maken met de planning of benodigde informatie met betrekking tot de medewerkers. De uitvoering kost een aantal uren op de zaterdagavond nadat alle bestanden binnen zijn. Vandaar dat dit deel is weggelaten in dit verslag.

2.2 Stroomschema

Hieronder volgt een stroomschema wat in grote lijnen weergeeft hoe de situatie in Groningen is met betrekking tot de productie van de **database** met vacatures. Op dit moment wordt de database wekelijks (op zaterdagavond) uitgeleverd. Op maandag wordt begonnen aan de nieuwe (lege) database. De pijlen geven weer hoe de databasebestanden het traject doorlopen.



2.3 Informatiestromen en gang van zaken

Binnen de organisatie zijn een aantal informatiestromen te onderscheiden. Aangezien voor dit project de relaties in en om Groningen alleen van belang zijn, worden ook alleen deze beschreven. Verder zijn er natuurlijk de gebruikelijke informatiestromen tussen directie en vestiging(en).

2.3.1 Gang van zaken in het algemeen

Er wordt vanuit gegaan dat alle medewerkers elke zaterdag werken, tenzij ze zelf aangeven dat ze niet kunnen. Voor degenen die ook door de week werken, geldt precies hetzelfde. Op een grote lijst met namen aan de muur van het kantoor wordt tot een aantal weken vooruit bijgehouden wie er wanneer is.

Alle medewerkers hebben de mogelijkheid zich aan of af te melden voor een bepaalde dag (meestal de zaterdag). Dit doen zij met een telefoontje, bij voorkeur zo vroeg mogelijk natuurlijk, maar soms ook op de zaterdagochtend. De reden wordt meestal alleen genoteerd als de afmelding erg laat of onverwacht was. Ze melden zich soms ook onverwacht weer aan.

2.3.2 Gang van zaken op zaterdag

Een aantal medewerkers weet van tevoren al wat ze moeten doen of waar ze moeten beginnen, daar is dan ook geen contact meer mee, tenzij er iets verandert. In het schema staat voor welke medewerkers dit geldt, maar deze kennis is bij de meeste vaste medewerkers ook al lang bekend.
Alle andere medewerkers krijgen 's morgens telefonisch te horen wat ze moeten doen. Zij bellen daarvoor tussen 9:00 en 10:00. De medewerkers in Groningen komen op hetzelfde tijdstip langs om hun werk op te halen.

Voor medewerkers in het land geldt: als ze klaar zijn sturen ze hun werk via het BBS door en bellen vervolgens om te vragen of het goed is aangekomen. Daarna geven zij de verdeling (klein, middel, groot) van hun werk door. Deze verdeling wordt opgeschreven om later verwerkt te worden voordat de lonen naar de boekhouder gaan.
De medewerkers in Groningen komen hun werk zelf op diskette (in tweevoud) terugbrengen en geven meteen hun verdeling door.
In principe is het de bedoeling dat iedereen tussen 15:00 en 16:00 klaar is en het werk heeft ingeleverd. Als er te weinig medewerkers of te veel vacatures zijn, mag men wat langer doorgaan. Dit is echter niet verplicht.

Als iemand verwacht dat hij of zij niet op tijd klaar zal zijn, moet er onmiddellijk gebeld worden en overlegd worden met de controleur wat er aan gedaan kan worden. Meestal betekent dit dat iemand wat langer door mag werken, of dat indien mogelijk een deel van het werk wordt teruggenomen. Al het werk wat terug komt, moet door de medewerkers op het kantoor in Groningen of Deventer gedaan worden.

In het begin van de middag komen nog een aantal kranten (welke dus geen ochtendkrant zijn), waardoor er rekening moet worden gehouden met extra benodigde capaciteit in de middag. Indien mogelijk wordt een deel hiervan aan medewerkers in het land gegeven.

Alle bestanden die binnenkomen via het BBS of diskette worden vrijwel onmiddellijk toegevoegd aan de database. Gedurende de dag en voornamelijk de avond wordt deze database gecontroleerd. Typefouten worden verbeterd, eventuele beginners worden extra gecontroleerd op principefouten. Tenslotte zijn er een aantal standaard controles nodig om een goede database af te leveren. Deze hangen echter niet af van de medewerkers.

2.3.3 Communicatie (informatiestromen)

De informatiestromen binnen het bedrijf, beperkt tot de communicatie tussen medewerkers en het kantoor in Groningen zijn het meest van belang. Hieronder volgt een overzicht van de aanwezige informatiestromen tussen de verschillende spelers.

Medewerkers in het land en in Groningen:
- Nieuwe medewerker meldt zich aan om een informatiepakket te ontvangen.
- Nieuwe medewerker besluit naar aanleiding van het pakket te gaan werken.
- Melden zich aan voor een bepaalde dag.
- Melden zich af voor een bepaalde dag.
- Geven aantallen door die ze willen doen op een dag.
- Geven aantallen door die ze gedaan hebben op een dag.

Vaste medewerkers (controleurs):
- Vragen een overzicht op van wie wat gaat doen (wordt gegenereerd).
- Wijzigen het schema door medewerkers opnieuw in te delen, aantallen te veranderen, etc.

Vaste medewerkers (administratie):
- Vragen loonberekening op.
- Verwerken gegevens van en naar het GAK.
- Verwerken gegevens van en naar het Ziekenfonds.

De meeste communicatie is per telefoon. Een medewerker vertelt een controleur wat hij of zij gedaan heeft en deze verwerkt de informatie in een van de schema's (op papier).

2.4 De toekomst

Op dit moment zijn er twee automatiseringsbedrijven bezig andere delen van het proces te automatiseren. Op relatief korte termijn (enige maanden) zal een begin van een nieuw systeem worden neergezet:
- Een volledig nieuwe invoerapplicatie (voor Windows)
- Een nieuwe collectie controleapplicaties (voor Windows)
- Vervanging van het BBS door een (betrouwbare) FTP applicatie en FTP-server.

Dit houdt in dat ook een deel van de communicatie zal veranderen. De belangrijkste verandering zal echter zijn op het gebied van de controle. De databasebestanden zullen op een andere manier binnenkomen. Daarna moeten ze ook op een andere manier verwerkt en gecontroleerd worden. Het principe, de kern van het systeem zal echter gehandhaafd worden.

3 De Opdracht

In het voorgaande ligt een opdracht voor het automatiseren van een onderdeel van het traject. Hieronder volgen eerst de gesignaleerde problemen en vervolgens de gewenste situatie en de opdracht voor het afstudeerproject dat daarmee samenhangt.

3.1 Problemen

- De verdeling-, registratie- en controletaken zijn verdeeld over meerdere (5) personen, die lang niet altijd van elkaar weten wat de ander al heeft gedaan en wat niet.
- Het is erg tijdrovend elke week weer uit te zoeken wie de beste kandidaat is voor een bepaalde taak.
- Vanwege het aantal medewerkers is het overzicht vaak zoek, het is nauwelijks mogelijk na te gaan of iedereen klaar is, nog bezig is, of helemaal niet is komen opdagen.
- Een wekelijks terugkerend karwei is het samenstellen van de lijst met medewerkers die op een dag (zaterdag) in die week werken.

3.2 Gewenste situatie

- Na het invoeren van de (handmatig) berekende aantallen moet automatisch een verdeling gegenereerd worden.
- Als het werk binnenkomt moeten de aantallen automatisch worden geregistreerd en vergeleken. Hierbij moet ook gedacht worden aan controle of het juiste werk wel is opgestuurd (niet dat van vorige week, of een bestand wat al eerder die dag was verstuurd). Eventueel moet hier ook een registratie bij komen voor de tijdsduur van de uitgevoerde werkzaamheden.
- Het moet mogelijk zijn om direct een volledig overzicht te krijgen van wat er nog moet gebeuren voordat de database als 'klaar' wordt beschouwd.
- Een centrale database moet bijhouden wie en wat door de controlegroep is gecontroleerd zodat bekend is wie wat heeft gedaan (dit wordt momenteel op papier of uit het hoofd gedaan).

3.3 De opdracht

- Analyseren welke informatiestromen er precies zijn vast te leggen en welke daarvan in een programma te verwerken zijn.
- Ontwerpen van een database die alle relevante (personeels)gegevens kan opslaan, bij voorkeur met historische gegevens.
- Ontwikkelen van een applicatie die alle relevante gegevens kan opslaan en weergeven, eventueel gekoppeld aan andere databases.
- Regels opstellen waarmee het programma in het beste geval alles kan plannen, en in het slechtste geval slechts enkele punten aan de controlegroep overlaat. Het programma moet nagaan of alles binnen is en (correct) gecontroleerd is.
- Vooraf moet een stappenplan (algemeen werkschema) worden gegenereerd dat die dag voltooid moet worden. Hierbij moet het programma zelf uitzoeken hoe de indeling van het personeel het beste gedaan kan worden in verband met capaciteit en locatie. Indien geen oplossing gevonden kan worden moet het probleem duidelijk aan de gebruiker getoond

worden en die de mogelijkheid geven zelf een oplossing in te voeren. Zo moet ook vooraf een capaciteitsprobleem gesignaleerd worden (met betrekking tot ochtend en/of middag).
- Het programma moet de mogelijkheid bieden de werkplanning op meerdere manieren weer te geven. Hierbij moet gedacht worden aan een planning per medewerker of per regio (of dagblad).
- Controleren of het uitgedeelde (geplande) werk overeen komt met het uiteindelijk ingeleverde werk. Als dit niet het geval is, inschatten wat er fout ging en dit rapporteren.
- Achteraf moet een resultaat geprint kunnen worden van alle relevante gegevens met betrekking tot het uitgevoerde proces en de uiteindelijke database.
- Als in de toekomst meer stappen geautomatiseerd worden (zoals het doorgeven wat het werk is dat gedaan moet worden, wat nu telefonisch wordt gedaan), is het de bedoeling dat deze informatie direct vanuit het programma gegenereerd wordt en beschikbaar wordt gemaakt voor de medewerkers.

Los hiervan zijn een aantal standaard zaken vereist die gebruikelijk zijn in een adres programma. Hierbij moet gedacht worden aan adressen voor mailings (of per medewerker voor opmerkingen na controle van deze medewerker).

De organisatie is op meerdere niveau's bezig is met automatisering van het proces. Een aantal programma's zullen op korte termijn (enige maanden) vervangen worden. Het programma moet dus flexibel zijn en makkelijk aan te passen aan een iets gewijzigde input of output.

4 Het ontwerp van de Database

Bij de gegevens die worden ingevoerd in de database moet onderscheid gemaakt worden tussen **directe** en **indirecte** invoer. De niet-vaste medewerkers (oproepkrachten, thuiswerkers) voeren indirect gegevens in, ze geven telefonisch of mondeling door wanneer ze wel en niet kunnen. Die data wordt rechtstreeks door een vaste medewerker ingevoerd. De vaste medewerkers (controleurs) hebben dus de rechten om alles direct in te voeren. Zij zijn ook verantwoordelijk voor het daadwerkelijk invoeren van de gegevens. Verder kunnen zij aanpassingen doen in het voorgestelde werkschema.
Als er dus sprake is van een actie die "gedaan" wordt door de oproepkracht, komt het er op neer dat deze de actie initieert, maar niet daadwerkelijk uitvoert.

4.1 Het Datamodel

Bovenstaande wordt op database niveau uitgewerkt in dit hoofdstuk. Eerst wordt de dataflow binnen de database beschreven. Daarna is het relationele model aan de beurt. Verderop wordt beschreven hoe de inhoud van de tabellen in de database er uit ziet.

4.1.1 Dataflow

Het is belangrijk een goed zicht te hebben op de dataflow binnen de organisatie, en het programma. Uit deze flow wordt namelijk de basis gehaald voor de gegevens in de database.

In de organisatie
De flow van informatie binnen de organisatie is al eerder uitgelegd. In het kort komt het er op neer dat er eerst een planning wordt gemaakt. Daarna worden de geplande taken uitgedeeld. Medewerkers die hun werk af hebben sturen de vacatures op en er wordt gecontroleerd of het ingeleverde werk overeenkomt met wat is uitgedeeld. Alle overige acties die ondernomen worden hebben voornamelijk betrekking op het inhoudelijke deel van vacatures en zijn daarom niet relevant in de database van het Planningsprogramma.

In de database
Een vergelijkbare flow moet ook in de database terug te vinden zijn. Om alle hiervoor benodigde stappen goed weer te geven, wordt de gang van zaken van achter naar voren uitgewerkt. Daarna kunnen de volgende stappen in de richting van het datamodel weer in chronologische volgorde genomen worden.

- Uiteindelijk is het de bedoeling dat er een registratie is van het werk dat de medewerkers gedaan hebben. Er moet gecontroleerd worden of het werk dat gedaan is overeenkomt met wat er is uitgedeeld.

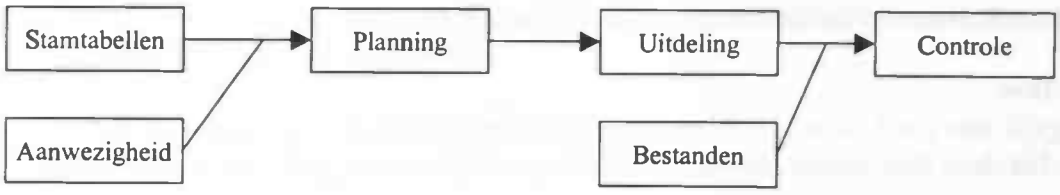
- Hiervoor is het noodzakelijk dat er bijgehouden wordt welke taken er aan medewerkers zijn uitgedeeld. Bij het uitdelen mag er geen medewerker of bron vergeten worden, om dit te garanderen moet er een planning gemaakt worden die alles dekt.

- In de planning wordt opgenomen welke medewerkers aanwezig zijn en welke taken zij kunnen doen. Tevens staat hier welke bronnen gedaan moeten worden. Deze gegevens worden uit de stamlijsten gehaald. Hierin staat welke medewerkers normaal gesproken aanwezig zijn en hoeveel werk ze kunnen doen. Daarnaast staat hier voor elke bron wanneer deze verschijnt en hoeveel vacatures er ongeveer in staan.

In de tabellen met Stamgegevens mogen alleen wijzigingen gemaakt worden als er een stamgegeven gewijzigd is. Dit betekent dat deze gegevens niet gemanipuleerd mogen worden bij het maken van een planning. Omdat de gegevens in de planning en uitdeling volledig gebaseerd zijn op de inhoud van de stamtabellen, moeten er verschillende lagen gecreëerd worden.

De stamtabellen bevatten de standaard gegevens over Bronnen, Medewerkers en hun Verdeling. Vanuit deze stamtabellen wordt de planning voor een dag gegenereerd en in de Planningstabel gezet. Daarna wordt hier invulling aan gegeven bij het uitdelen. Dit is terug te vinden in de Uitdelingstabel. De relatie met de planning wordt hierin bijgehouden.

Medewerkers sturen hun bestanden op, welke in de Bestandstabel worden opgenomen. In de Controletabel wordt tenslotte bijgehouden wat de verschillen en overeenkomsten zijn tussen de uitdeling en de ontvangen bestanden.



4.1.2 Gevolgen voor de database

De stamgegevens zoals bronnen en medewerkers zijn constant te noemen. Voor de data in de Aanwezigheid geldt een slechts beperkte levensduur (tot de week na de dag waarop de wijziging van kracht was). De overige tabellen (Planning, Uitdeling, Bestanden en Controle) zijn slechts van belang in de week waarin ze gemaakt worden.

De Planning en de tabellen die daarna komen bevatten informatie die is afgeleid uit de stamtabellen (aangevuld met de gegevens in Aanwezigheid). Hieruit zou de gedachte kunnen ontstaan dat deze gegevens als afgeleide gegevens opgeslagen kunnen worden, welke met eventuele queries opgevraagd kunnen worden.

Deze gedachte is echter onjuist, en wel om de volgende redenen:

1. De planning is altijd vooraf (met een schatting van de aantallen), de uitdeling is al gebaseerd op een iets meer actuele situatie. De controle is pas mogelijk op het moment dat de werkelijkheid goed in kaart is gebracht.
2. De fysieke dataflow (van vacatures) brengt veel actuele gegevens met zich mee. Deze actuele gegevens zijn niet in de database te verwerken als daar niet een aparte plaats voor is. Met deze actuele gegevens moeten later (bijvoorbeeld bij de controle) weer nieuwe gegevens worden gegenereerd.
3. De gegevens die uit de stamgegevens worden gegenereerd kunnen op meerdere manieren en tijden gewijzigd worden, waardoor er telkens met een andere set gewerkt moet worden. Deze set kan niet een afgeleide zijn van de stamgegevens, omdat de mogelijkheid bestaat dat er inmiddels meerdere veranderingen in gemaakt zijn.

De manipulatie van de gegevens zodat deze de actuele situatie goed nabootsen kan zeer goed door gebruik te maken van meerdere lagen. In iedere laag komt de nagebootste situatie een

stap dichter bij de realiteit. Het opslaan in een database met maar één laag zou een onverstandige keuze, en een vrijwel onmogelijke opgave zijn.

4.1.3 Relationeel model

Om een en ander goed te kunnen beschrijven, is het noodzakelijk een eenvoudig relationeel model te maken. In dit model worden de verschillende objecten (abstract gezien) en de relaties daartussen beschreven. Deze beschrijving bepaalt ook direct de keuze voor de tabellen en hun inhoud.

4.1.3.1 Entiteiten

De basis van de zogenaamde entiteiten wordt gevormd door het definiëren van objecten. Elk object heeft een of meerdere attributen. Er wordt nu (nog) geen onderscheid gemaakt tussen abstracte en concrete objecten. Hieronder volgt een opsomming van de objecten en hun attributen.

Het zal inmiddels duidelijk zijn dat alles draait om de zogenaamde *Taak* die medewerkers krijgen (gepland, uitgedeeld). Dit object staat dus centraal in het gehele model.

Bron

- Naam
- Verschijningsdatum
- Gemiddeld aantal vacatures

Medewerker

- Persoonsgegevens (naam en adres)
- Werkplek (thuis of op kantoor)
- Beschikbaarheid

Taak

- Medewerker
- Bron
- Hoeveelheid vacatures
- Datum

Planning

- PlanningID
- Taken

Uitdeling

- PlanningID
- UitdelingID
- Taken

Controle

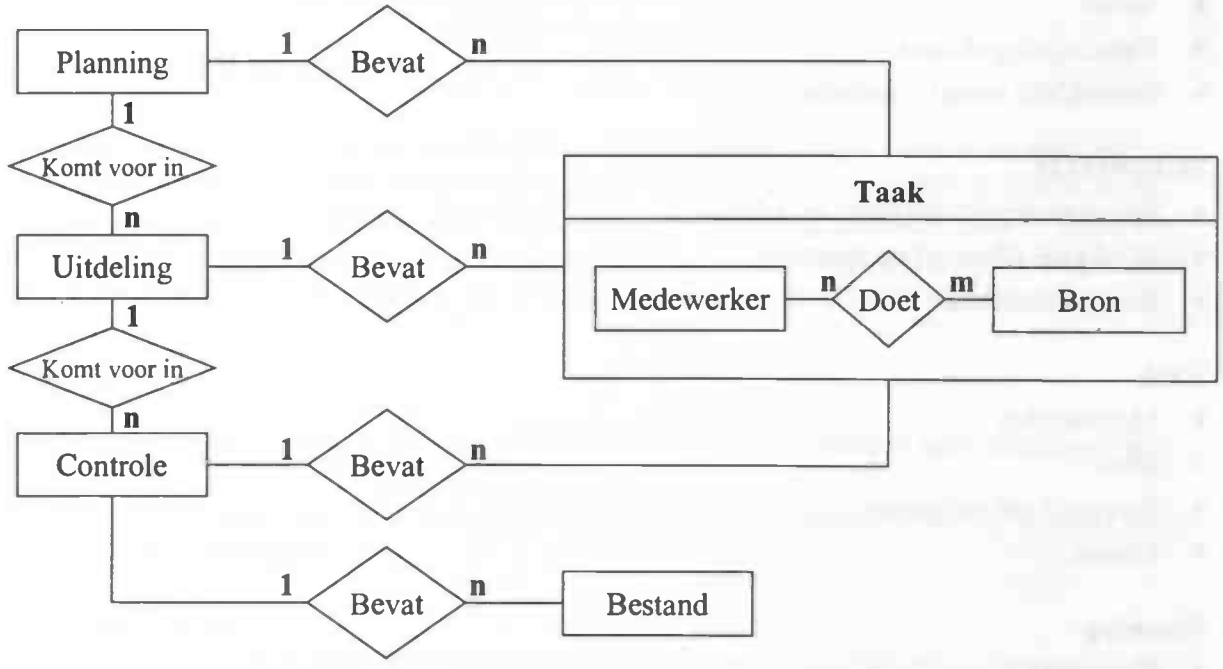
- UitdelingID
- InfoID
- Medewerker

Bestand

- InfoId
- Bestandsnaam
- Medewerker
- Bronnen
- Aantallen per bron

4.1.3.2 Relaties

De relaties tussen genoemde objecten en hun attributen geven weer hoe de database intern in elkaar moet zitten. Deze relaties zijn in onderstaand figuur weergegeven en vormen samen met de entiteiten het relationele model.



4.2 De eerste opzet

In het begin is gekozen voor de naïeve aanpak: een aantal basistabellen ontwerpen en al naar gelang uitbreiden. De voordelen van deze methode zijn onder andere de mogelijkheid om snel aan het werk te gaan in de Delphi applicatie en de mogelijkheid tot het direct testen van de al aanwezige ideeën. De aanwezigheid van deze ideeën is niet anders dan logisch omdat het afstudeerproject grotendeels door mijzelf geformuleerd is en ik in die tijd ook al heb nagedacht over hoe de gegevens moesten worden opgeslagen. Een laatste voordeel lag in het feit dat er op deze manier zo gefaseerd mogelijk gewerkt kon worden. Ieder deel van het programma moest een ander deel van de database gebruiken.

Aangezien het programma moet samenwerken met andere programma's moet er ook een zekere consistentie bestaan tussen de database van de bestaande programma's en deze database. Dit was de aanzet om persoonsgegevens en de bronnenlijst rechtstreeks over te nemen uit het bestaande controleprogramma.

De eerste tabellen waren vrij eenvoudig, vrijwel alles wat nodig was stond in een tabel. Na het ontstaan van overige tabellen zijn deze eerste versies wel ingewikkelder geworden. Een aantal gegevens kwam al vrij snel meerdere keren voor. Deze velden zijn uiteraard vrij snel verzameld in een aparte tabel.

4.2.1 De eerste implementatie

Al in een vroeg stadium zijn de tabellen van papier geïmplementeerd in het **DB-III** formaat. Dit levert platte tabellen op die elk in een apart bestand staan. Met eenvoudige programma's is het prima mogelijk deze bestanden te lezen of te corrigeren zonder dat hier een bepaalde database-engine voor nodig is.

Gekozen is voor een toegang door middel van de Microsoft ADO componenten. Deze zijn in Delphi 5 geïntegreerd en dus zonder verdere installatie en documentatie te gebruiken. Het alternatief (de door Borland meegeleverde BDE) moet expliciet op elke computer geïnstalleerd worden, terwijl de ADO componenten bij Windows 98 en hoger al bij de installatie automatisch geïnstalleerd worden.

Het eerste nadeel van de keuze voor ADO komt naar voren als blijkt dat de benodigde SQL-queries in een BDE-omgeving wel met een speciale editor gemaakt kunnen worden, maar dit in de ADO-omgeving niet het geval is. Programmeren in de ene omgeving en omzetten naar de andere is niet mogelijk, de gegenereerde code bevat BDE-specifieke commando's welke niet in andere omgevingen werken. Het gevolg is dat de SQL-queries in hun geheel zonder hulpmiddelen geprogrammeerd moeten worden. Naar aanleiding van deze situatie wordt afgesproken alle queries in *standaard* SQL te schrijven.

Nadat er inmiddels 2 maanden gewerkt is met de DB-III tabellen, blijkt dat het toch een groot probleem is een goede connectie te maken met de standaard drivers. De initialisatie blijkt taalgevoelig te zijn (op Engelstalige Windows machines werkt het anders dan op Nederlandstalige Windows machines). De oplossing is na enige tijd gevonden door rechtstreeks de naam van de onderliggende driver te gebruiken.

Tijdens een demonstratie op een (standaard) computer waar de ontwikkelomgeving niet is geïnstalleerd blijkt dat er na deze aanpassingen nog steeds problemen zijn. Na vele zoekacties op Internet begint de oorzaak van de problemen door te schemeren. Microsoft heeft in de MDAC (componenten met daarin ADO) vanaf versie 2.1 de benodigde DB-drivers zo aangepast dat deze alleen nog maar een DB-tabel kunnen lezen. Voor het schrijven is vanaf die versie de BDE nodig. Tevens zijn er nog wat geruchten over mogelijke datacorruptie die kan voorkomen als er DB of FoxPro tabellen worden benaderd met als doel het wijzigen van data.

4.2.2 De tweede poging

Om verdere problemen zo veel mogelijk te voorkomen is ervoor gekozen de database in Microsoft Access te implementeren. Access is net als ADO een product van Microsoft zelf en de garanties met betrekking tot de ondersteuning zijn daardoor beter. Aangezien Access alleen

maar voor de opslag (de tabellen zelf) gebruikt wordt, verandert dit niets aan de geplande situatie waarin de applicatie in Delphi geprogrammeerd wordt. De bestaande tabellen zouden gewoon geïmporteerd kunnen worden. Na ongeveer een dag bleek dat Access geen tabellen kan importeren als de bestandsnaam langer is dan 8 tekens, hoewel deze wel te selecteren zijn. Laat nu net het eerste bestand een naam hebben van 9 tekens en de foutmelding "object not found" zeer weinig zeggen over de werkelijke situatie.

Daarna verliep de implementatie in Access eigenlijk vlekkeloos. Een bijkomend voordeel is dat de queries in Access zelf opgeslagen kunnen worden en daar ook ontworpen kunnen worden. De (visuele) hulpmiddelen maken het ook mogelijk de gewenste *views* te maken (dit zijn eigenlijk opgeslagen queries).

Een niet verwacht neveneffect binnen Access bracht een ander probleem aan het licht. De unieke code die aan iedere bron in de bestaande bronnenlijst wordt toegekend, blijkt niet zo uniek te zijn als verwacht. Deze fouten zijn verholpen in het controleprogramma dat de bronnenlijst gecreëerd heeft, waarna de lijst opnieuw geïmporteerd kon worden.

Na de relatief eenvoudige aanpassingen in het prototype van de applicatie werkt deze voortaan met de Access database. De eerste test op een machine anders dan de ontwikkelmachine is voor het eerst succesvol.

4.3 Uiteindelijke database

Hieronder volgt de beschrijving van de uiteindelijke database, zoals deze in de applicatie gebruikt wordt. Er zijn enkele verschillen tussen wat hier beschreven staat en wat er in de Access database (**"Pcc.mdb"**) te zien is. Hier is alleen rekening gehouden met de delen die daadwerkelijk geïmplementeerd zijn. In de technische handleiding van de applicatie worden ook de overige tabellen beschreven die in de toekomst gebruikt kunnen worden voor uitbreidingen.

4.3.1 Tabelindeling en inhoud

De database is opgebouwd uit een aantal tabellen. Hieronder wordt per tabel uitgelegd waar deze voor nodig is en welke gegevens er in staan.
De gegevens die (nog) niet in te vullen zijn, maar waarvan de velden al wel in de tabel staan, worden *cursief* weergegeven.

Persoonsgegevens
In deze tabel worden de gegevens van alle thuiswerkers en de (niet vaste) medewerkers op kantoor bijgehouden. De *code* van de medewerker wordt ook wel de *samenvatcode* genoemd. Dit is een 3-letterige code waarin een deel van de naam van de medewerker verwerkt is. Deze code wordt als bestandsextensie gebruikt om bestanden met vacatures van de medewerkers te herkennen. In de vacaturedatabase wordt deze code bij elke vacature bewaard, zodat later terug te vinden is door wie de vacature ingevoerd is.

Veldnaam	Inhoud	Omschrijving
MDW_ID	Uniek, Int(5)	Uniek ID wat aan een medewerker wordt toegekend zodra deze wordt toegevoegd in de database.
CODE	Char(3)	Unieke code die in de vacaturedatabase gebruikt wordt voor een medewerker (deze kan hergebruikt worden).
VOORNAAM	Char(35)	Voornaam van de medewerker.
ACHTERNAAM	Char(35)	Voorvoegsels en achternaam van de medewerker.

ADRES	Char(35)	Straatnaam en –nummer
POSTCODE	Char(7)	Postcode (inclusief spatie tussen de cijfers en letters)
WOONPLAATS	Char(35)	Woonplaats
EMAIL	Char(60)	Emailadres
BANKGIRO	Char(10)	Bank- of girorekening nummer
GEBDATUM	Date	Geboortedatum van de medewerker
GESLACHT	Char(1)	Geslacht van de medewerker ("M" of "V")
SOFINUMMER	Char(15)	Sofinummer van de medewerker
PERSNUMMER	Int(3)	Personeelsnummer wat door de boekhouder aan de medewerker is toegekend.
TARIEFGR	Int(1)	Tariefgroep (belasting) van de medewerker.
UURLOON	Int(4)	Uurloon in centen.
GAKAANMELD	Date	Datum waarop de medewerker is aangemeld bij het GAK.
GAKAFMELD	Date	Datum waarop de medewerker is afgemeld bij het GAK.
ZKFAANMELD	Date	Datum waarop de medewerker is aangemeld bij het Ziekenfonds.
ZKFAFMELD	Date	Datum waarop de medewerker is afgemeld bij het Ziekenfonds.
STARTDATUM	Date	Datum waarop de medewerker is begonnen
STOPDATUM	Date	Datum waarop de medewerker is gestopt of gaat stoppen.
NAARKANT	Int(1)	Geeft aan of een medewerker naar een vestiging (kantoor) gaat om vacatures op te halen. (0 is niet, 1…9 zijn vestigingen)
OPKANTOOR	Logic	Waar als de medewerker (bij voorkeur) op het kantoor in Groningen werkt.
ACTIEF	Logic	Waar als de medewerker nog actief is (en dus niet waar als de medewerker langere tijd niet werkt of zelfs helemaal gestopt is).
OPMERKING	Char(250)	Eventuele extra opmerkingen bij deze medewerker

Telefoonnummers

De meeste medewerkers hebben meerdere telefoonnummers. Zelfs al voordat iedereen een mobiele telefoon had, waren er in het bedrijf medewerkers met maar liefst 3 nummers. Veel van de thuiswerkers zijn studenten die in het weekend bij hun ouders werken.

Veldnaam	**Inhoud**	**Omschrijving**
MDW_ID	Int(5)	Medewerker ID
TELEFOONNR	Char(15)	Telefoonnummer (inclusief streepje, maar zonder spaties)

Vasteverdeling

De *vaste* of ook wel *standaard* verdeling moet beschouwd worden als een uitgangspunt voor de actuele verdeling. In deze tabel wordt van een aantal medewerkers bijgehouden hoeveel ze over het algemeen kunnen doen. Als ze *vast* op een bron staan, betekent dit dat ze zonder tegenbericht gewoon beginnen of in ieder geval zeker weten dat ze die bron 's morgens kunnen halen. Als bij een verdeling *veilig* waar is, dan begint deze medewerker zeker in de aangegeven bron.

Hoe meer gegevens er in deze tabel staan, hoe beter. Als er voor een medewerker geen gegevens zijn ingevuld voor een bepaalde dag, geeft de applicatie een aantal standaardwaarden bij het plannen. Deze standaardwaarden zijn uiteraard minder gewenst dan een vooraf geplande hoeveelheid en bronverdeling.

Veldnaam	**Inhoud**	**Omschrijving**
STDVERD_ID	Uniek, Int(10)	Unieke ID behorend bij deze vaste/standaard verdeling
MDW_ID	Int(5)	Medewerker ID
BRON_ID	Int(4)	Optionele Bron ID. Als er geen vaste verdeling is kan hier een voorkeursbron staan.
DAGVDWEEK	Int(1)	Dag van de week waarop de medewerker deze werkzaamheden standaard uitvoert (1 is zondag, 7 is zaterdag)

PRIORITEIT	Int(1)	Prioriteit die deze medewerker op deze bron heeft, 1 is hoogste prioriteit, 9 laagste.
KATERN	Char(60)	Optioneel, aanwijzing voor welk deel (welk katern) van de krant normaal gesproken gereserveerd. Dit is een indicatie voor de gebruiker, niet voor het programma.
VEILIG	Logic	Waar als deze medewerker zonder contact gewoon begint en dit katern of deze bron af maakt (wanneer de situatie dus "veilig" is).
VAST	Logic	Is waar als deze medewerker standaard de opgegeven bron doet.
AANTALMIN	Int(3)	Minimum aantal wat normaal gesproken gedaan wordt.
AANTALMAX	Int(3)	Maximum aantal wat normaal gesproken gedaan wordt.

Aanwezigheid

De regel binnen het bedrijf is dat iemand elke zaterdag werkt, tenzij hij of zij zich afmeldt. Dezelfde regel wordt ook in de database gehanteerd: een medewerker staat alleen in de tabel aanwezigheid als deze zich heeft afgemeld. Eigenlijk zou "Afwezigheid" een betere naam zijn voor deze tabel.

Veldnaam	Inhoud	Omschrijving
MDW_ID	Uniek, Int(5)	Medewerker ID
DATUM	Date	Datum waarvoor de medewerker zich heeft afgemeld.
AFGEMELD	Logic	Alleen niet waar als de medewerker zich later alsnog weer heeft aangemeld.
AFMELDDATE	Date	Datum (en tijd) waarop de afmelding plaatsvond.
REDEN	Char(64)	Optionele reden van afmelding.

Bronnenlijst

In de bronnenlijst wordt de naam en het nummer van alle bronnen opgeslagen. De namen worden overgenomen uit een bestaande lijst. De verschijningsdatum (dag van de week), verkrijgbaarheid en of deze bron normaal gesproken gedaan wordt of niet is extra informatie die wordt toegevoegd aan de al aanwezige lijst. Voor bronnen die (alleen) op kantoor te verkrijgen zijn, zoals tijdschriften, is er de mogelijkheid aan te geven op welke vestiging deze bron bezorgd wordt.

Veldnaam	Inhoud	Omschrijving
BRON_ID	Uniek, Int(4)	ID dat aan de bron wordt toegekend bij het toevoegen van een bronnaam.
BRONNAAM	Char(60)	Naam van de bron
DAGVDWEEK	Int(1)	Dag van de week waarop de bron verschijnt. (0=dagelijks, 1=zondag, 7=zaterdag).
PAUZEWEEK	Int(1)	Aantal weken dat tussen verschijning van de bron zit (0=wekelijks, 1=2-wekelijks, ...)
GEDAAN	Logic	Waar als de bron gedaan wordt, niet waar als de bron niet (meer) verkrijgbaar is, of niet in de planning moet voorkomen (dit wil niet zeggen dat de bron niet voor kan komen!).
VACATURES	Int(4)	(Gemiddeld) aantal vacatures wat in de bron te verwachten is.
LANDELIJK	Logic	Waar als de bron landelijk (overal) verkrijgbaar is. Dit geldt dus alleen voor landelijke (dag)bladen, en niet voor huis-aan-huis, regionale of weekbladen (die op kantoor komen).
OPKANTOOR	Int(1)	Geeft aan op welk kantoor de bron (standaard) komt (met de post). 0 als de bron in het geheel niet op kantoor komt.

Planning

De planning kan gezien worden als een relatietabel tussen de medewerkers (Persoonsgegevens) en de Bronnenlijst, die gevuld wordt met gegevens uit de Vasteverdeling

(gegenereerd). De datum van de bron wordt hierin bijgehouden omdat deze kan afwijken van de datum die de bron normaal gesproken heeft. Aangezien de planning per week wordt bepaald, zijn ook het weeknummer en het jaar in de tabel opgenomen.

Veldnaam	Inhoud	Omschrijving
PLAN_ID	Uniek, Int(10)	ID om in de uitdeling te kunnen zien waar deze bij hoorde.
STDVERD_ID	Int(10)	Geeft aan bij welke vaste verdeling deze planning hoort.
MDW_ID	Int(5)	Medewerker ID
GENERATED	Logic	Is waar als de planning van dit werk gegenereerd is (en dus niet waar als de planning door de gebruiker is ingevuld).
DATUMBRON	Date	Datum van de bron die gepland is.
GEPLANDMIN	Int(4)	Minimum aantal dat gepland is.
GEPLANDMAX	Int(4)	Maximum aantal dat gepland is.
BRON_ID	Int(4)	Unieke Bron ID van de bron die gepland is.
DAGVDWEEK	Int(1)	Dag waarop deze hoeveelheid gepland is.
VEILIG	Logic	Is waar als deze verdeling afgesproken en dus veilig is.
WEEKNR	Int(2)	Weeknummer (volgens officiële weeknummering)
JAAR	Int(4)	Jaar waarin de week valt

Uitdeling

In de tabel Uitdeling staan zo ongeveer dezelfde gegevens als in de tabel Planning. Zodra een deel van het geplande werk daadwerkelijk wordt uitgedeeld, worden de gegevens van de planning overgenomen in deze tabel. Eventueel kan er aanvullende informatie over het gebruik van snellijsten worden opgenomen. Het is ook mogelijk onverwachte (niet geplande) uitdelingen toe te voegen.

Veldnaam	Inhoud	Omschrijving
UITGED_ID	Uniek, Int(10)	ID om bij met gedane werk te kunnen koppelen.
PLAN_ID	Int(10)	Planning ID waarop deze uitdeling is gebaseerd, -1 indien er geen planning bij hoort.
MDW_ID	Int(5)	Medewerker ID
DATUMBRON	Date	Datum van de bron die uitgedeeld is.
UITGEDMIN	Int(4)	Minimum aantal dat uitgedeeld is.
UITGEDMAX	Int(4)	Maximum aantal dat uitgedeeld is.
BRON_ID	Int(4)	Unieke Bron ID van de bron die uitgedeeld is.
SNELAANTAL	Int(4)	Aantal vacatures wat vanuit een snellijst is gedaan.
DAGVDWEEK	Int(1)	Dag waarop deze hoeveelheid uitgedeeld staat.
WEEKNR	Int(2)	Weeknummer (volgens officiële weeknummering)
JAAR	Int(4)	Jaar waarin de week valt

Bestandsinfo

In deze tabel wordt alle informatie opgeslagen die wordt gehaald uit de binnenkomende bestanden. De bestandsdatum is nodig om te kunnen zien of een bestand al eerder is binnengekomen. Nadat een bestand geïmporteerd wordt, is het de bedoeling in de tabel aan te geven dat dit gebeurd is. Een bestandsnaam kan meerdere keren voorkomen, omdat in één bestand meerdere verschillende bronnen kunnen staan.

Veldnaam	Inhoud	Omschrijving
INFO_ID	Int(10)	Autonummering van ID's
BESTAND	Char(255)	Bestandsnaam.
BESTANDSDATUM	Date	Datum en tijd van het bestand.
MDW_ID	Int(5)	ID van de medewerker.
BRON_ID	Int(4)	ID van de bron die gedaan is.
DATUM	Date	Datum van de bron die gedaan is.

AANTAL	Int(4)	Aantal dat gedaan is.
SNEL	Int(4)	Aantal dat gedaan is uit snellijsten.
IMPORT	Logic	Is waar als het bestand is geïmporteerd

Controle
De controletabel bevat de koppeling tussen wat uitgedeeld is en wat teruggekomen is (tabel Uitdeling en tabel Bestandsinfo). Per combinatie kan aangegeven worden wat de status is en wat eventuele fouten zijn.

Veldnaam	Inhoud	Omschrijving
CONTROLE_ID	Int(10)	Autonummering van de ID's
UITGED_ID	Int(10)	Uitdeling ID (mag ook -1 zijn)
INFO_ID	Int(10)	BestandsInfo ID (mag ook -1 zijn)
MDW_ID	Int(5)	ID van medewerker die in dit bestand staat.
STATUS	Int(6)	Status van deze match (bitveld)

4.3.2 Views en hun inhoud
In Access zijn twee queries gemaakt die het gemakkelijker maken om bepaalde gegevens op te vragen. Deze views voorkomen ook dat voor- en achternamen elke keer opnieuw aan elkaar gekoppeld moeten worden wanneer de volledige naam nodig is.

Namenlijst
Bevat de ID van de medewerker, zijn/haar voornaam, achternaam en code.

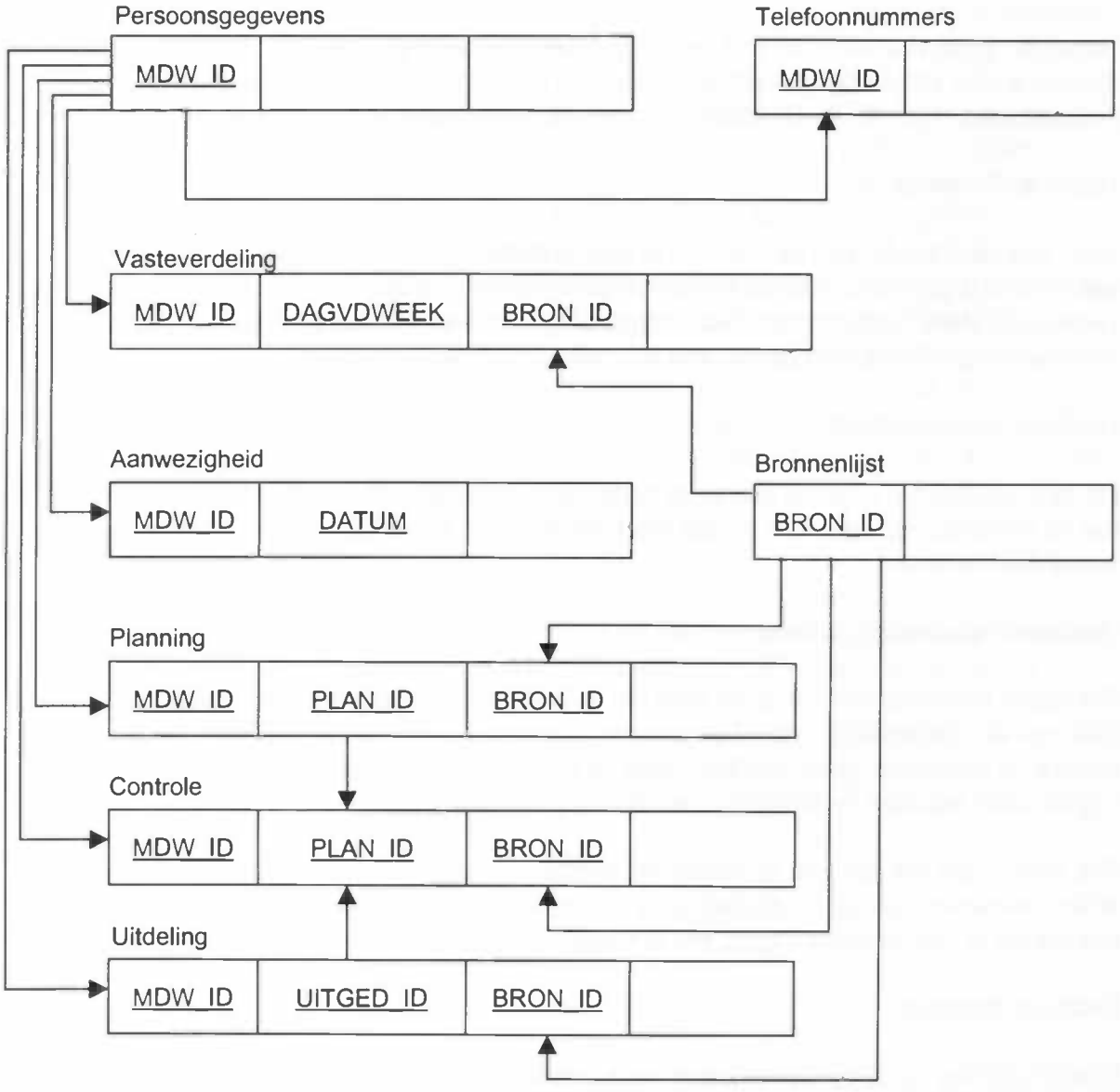
Veldnaam	Inhoud	Omschrijving
MDW_ID	Int(5)	ID van de medewerker.
NAAMENCODE	Char(..)	Voornaam, achternaam en code van de medewerker
CODE	Char(3)	Code van de medewerker.

Importlijst
Omdat in de tabel Bestandsinfo een bestandsnaam meerdere keren voor kan komen omdat er meerdere bronnen in kunnen staan, is in deze tabel per bestand het totaal aantal vacatures opgenomen.

Veldnaam	Inhoud	Omschrijving
BESTAND	Char(255)	Naam van het bestand.
CODE	Char(3)	Code van de medewerker.
IMPORT	Logic	Of het bestand geïmporteerd is of niet.
TOTAAL	Int(..)	Totaal van vacatures dat in het bestand staat.

4.3.3 Datadiagram

De relaties tussen de tabellen worden in onderstaand datadiagram nog iets verduidelijkt.



4.4 Historische gegevens in de database

Een aantal gegevens in de database kan en/of moet bewaard worden. Andere gegevens kunnen na gebruik en rapportage als overbodig worden beschouwd. Hieronder volgt een overzicht van de delen van de database die bewaard kunnen worden, en de reden waarom dit wel of niet gebeurt.

Database/Planning

In de database zijn uiteraard de gegevens van de planning van de huidige week nodig. Om de lonen te kunnen berekenen zijn echter ook de gegevens van vorige weken (de hele maand)

nodig. Ook voor het plannen zijn deze gegevens nodig omdat aan de hand van aantallen van een vorige week, het overgebleven werk voor de huidige week kan worden bepaald.

Database/Uitdeling

Hetzelfde geldt voor de tabel met gegevens over de uitdeling. Later kan bijvoorbeeld gekeken worden welke verschillen er optreden tussen de planning (al dan niet gemaakt door het programma zelf) en de daadwerkelijke uitvoering daarvan.

Database/Controle

Voor alle tabellen die met de controle te maken hebben geldt dat ze niet meer nodig zijn als een week is afgesloten. Aan het einde van de week worden de vacatures verwerkt en wordt de vacaturedatabase leeggemaakt. Deze tabellen worden dan ook aan het begin van de volgende week automatisch schoongemaakt (na een waarschuwing).

Database/Aanwezigheid

De aanwezigheid is achteraf niet meer nodig, behalve misschien ter controle. Hierbij is het dus de bedoeling de gegevens na een week automatisch uit de database te verwijderen en in een archief te zetten.

Database/Personeelsgegevens

Het aantal medewerkers kan in de loop der tijd behoorlijk oplopen. Vanwege de soms korte duur van de overeenkomst, kan het gewenst zijn medewerkers te verwijderen. Medewerkers kunnen op non-actief gezet worden, zodat ze niet meer in de lijsten voorkomen. Er wordt bijgehouden wanneer medewerkers worden uitgeschreven.

Een aantal keer per jaar kan de medewerkerdatabase opgeschoond worden door alle non-actieve medewerkers in een archief op te zetten. Hierbij moet de uitschrijvingsdatum zeker een maand in het verleden liggen, om te voorkomen dat namen zomaar worden geschrapt.

Database/Bronnen

In tegenstelling tot de medewerkers is het bij de bronnenlijst niet nodig delen uit de database te bewaren. Als de bron niet meer in het Controleprogramma voorkomt, is deze ook zeker niet meer nodig in het Planningsprogramma.

4.5 Opmerkingen over de implementatie

Door de tegenslagen, veroorzaakt door ADO en Access onderdelen is er erg veel tijd verloren gegaan. In een aantal gevallen is er een mentaliteit ontstaan die neerkomt op "als het maar werkt". Dit heeft tot gevolg gehad dat er geen tijd meer is geweest om gebruik te maken van specifieke Access features. Hierbij moet vooral gedacht worden aan bepaalde relaties en integriteitsregels. De implementatie hiervan is overigens Access-specifiek, en zou voor een ervaren Accessprogrammeur geen problemen moeten opleveren.

In het ontwerp wordt regelmatig een *boolean* gebruikt. In de gebruikte queries kan hier eenvoudig gebruikt van gemaakt worden door "=true" of "=false" in het SQL statement in te

vullen. In Delphi worden boolean waarden echter als '-1' (true) of '0' (false) teruggegeven. Dit heeft tot gevolg dat deze waarden niet als boolean kunnen worden opgevraagd, maar als integer gezien moeten worden. In Delphi is het dus noodzakelijk de boolean waarde terug te krijgen door middel van een statement als "waarde=-1".
De verwarring die hierdoor kan ontstaan is dus niet het gevolg van de opzet van de database, maar een neveneffect van de keuze voor Access.

Een soortgelijk probleem is ook aanwezig in combinatie met ADO. Bij het opvragen van gegevens uit de database wordt een soort van caching gebruikt. Deze caching is echter niet correct geïmplementeerd. Als er in de database gegevens van een tabel geupdate worden (of nieuwe records worden toegevoegd), kan het gebeuren dat bij het opnieuw openen van een query deze gegevens (nog) niet zichtbaar zijn.
De enige oplossing is alle connecties met de database verbreken na een update, en deze weer opnieuw te openen. Dit veroorzaakt echter een onacceptabel lange wachttijd bij elke wijziging. In meerdere discussiegroepen is te lezen dat meer programmeurs hier last van hebben. Alle daar aangedragen oplossingen werken niet, of niet voldoende. In de implementatie is met dit probleem zo veel mogelijk rekening gehouden door voor de weergave op het scherm gebruik te maken van in het geheugen geladen lijsten, in plaats van rechtstreeks informatie uit de database weer te geven.

4.6 Beveiliging

De database krijgt binnen de organisatie een plaats op het netwerk die alleen door de daartoe bevoegde medewerkers benaderd kan worden. Dit eenvoudige principe is ruimschoots voldoende om de database te beveiligen.

5 Gebruikte algoritmes

Elke week is een schema nodig waarin staat welke bronnen door welke medewerkers gedaan worden. Hiervoor is een werkverdeling (planning) nodig. Het verdelen van het werk bestaat uit drie onderdelen:

- Van tevoren wordt er een **planning** gemaakt. Deze planning komt tot stand door het bekijken van alle bronnen die gedaan worden. Vervolgens worden beschikbare medewerkers geselecteerd die een (deel van een) bron moeten doen.
- Op het moment dat het werk daadwerkelijk verdeeld wordt, is het noodzakelijk bij te houden welke bron en hoeveel er uitgedeeld is aan een medewerker. Dit wordt de **uitdeling** genoemd.
- Tenslotte wordt er **gecontroleerd** of het werk wat ingeleverd wordt overeenkomt met het werk wat uitgedeeld is. Hiervoor is de interne registratie noodzakelijk.

Hieronder worden de algoritmes van de verschillende onderdelen verder uitgewerkt. De uitdeling is echter een taak van de gebruiker van het programma. Hier is dus ook alleen een weergave van de actuele situatie aanwezig, en geen algoritme.

5.1 Algoritme voor planning

De planning van het werk is afhankelijk van de dag van de week. Meer dan de helft van het werk wordt op zaterdag gedaan. Donderdag en vrijdag komen op de tweede plaats wat betreft het aantal vacatures. Het algoritme is echter in alle gevallen identiek.

5.1.1 Handmatige versie

In een kort interview met de verantwoordelijke voor het maken van de planning zijn een aantal punten naar voren gekomen die moeten worden overgenomen door het planningsalgoritme.

De planning voor de zaterdag gaat handmatig als volgt:

1. Aanwezigheid bepalen door middel van de lijst. Alle namen van aanwezige medewerkers worden opgeschreven.
2. Standaardschema (een Word document) openen:
   - **Regionale** bronnen nalopen en daar **vaste** personen op zetten.
     Als die er niet zijn, een reserve er bij op zetten.
   - **Landelijke** bronnen niet vaste, maar **voorkeur** personen op zetten, waarbij de aantallen per katern worden bepaald en geprobeerd wordt deze "vol" te krijgen. Bij voorkeur medewerkers niet te veel laten wisselen tussen bronnen.
3. De "reserves" die niet nodig waren bij een regionale bron worden bij een landelijke bron toegevoegd.

De planning voor door de week is minder strak, er zijn dan namelijk veel minder vacatures te verdelen. De medewerkers komen langs of krijgen een portie opgestuurd. Hierbij wordt alles wat op dat moment voorradig is zo snel mogelijk uitgedeeld.

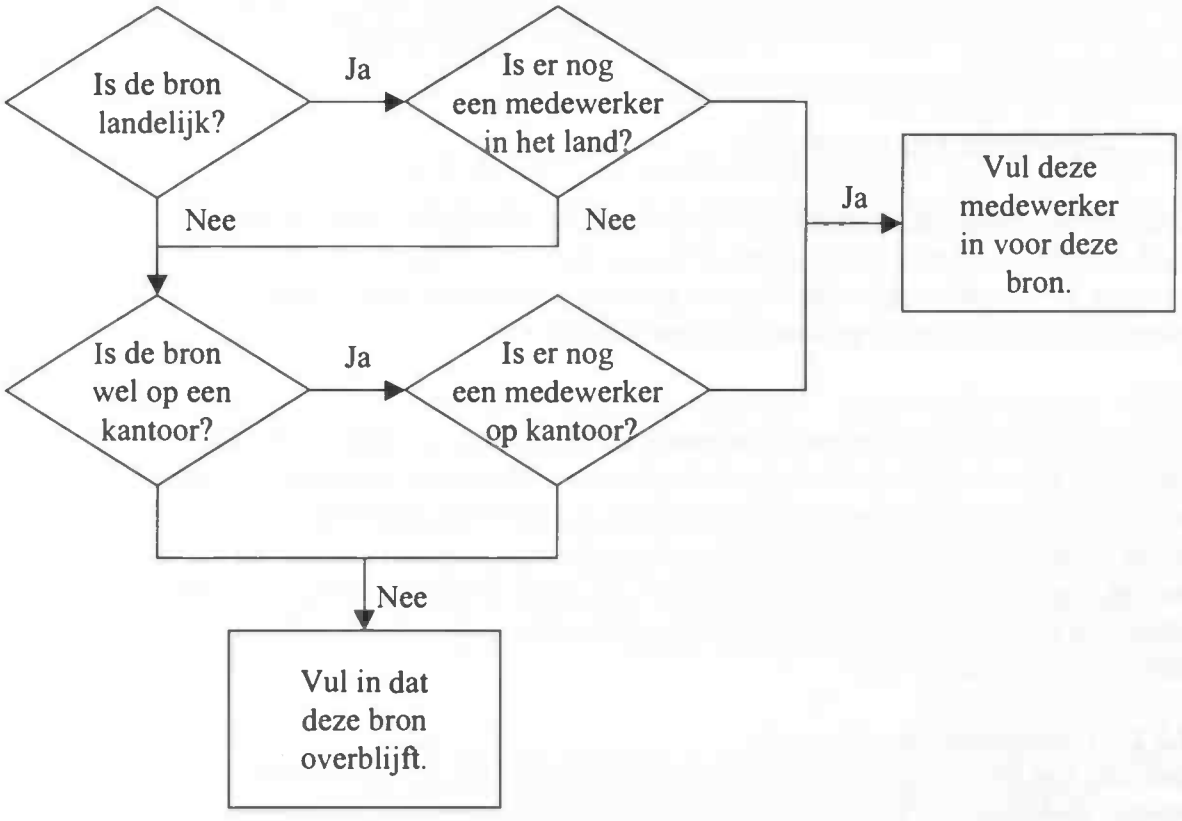
5.1.2 Geïmplementeerde versie

Het algoritme dat bovenstaande handelingen vastlegt moest zo dicht mogelijk bij de oorspronkelijke methode blijven. Ten eerste omdat de methode zelf goed werkt, en ten tweede omdat dit eenvoudiger te integreren is in de organisatie.

Het plannen is te verdelen in de volgende stappen:

1. Maak een lijst van alle medewerkers die op de gewenste dag aanwezig zijn en niet vast op een bron staan ingedeeld (de *Beschikbare Medewerkers* lijst). Houd bij waar de medewerker zich bevindt (in het land of op een vestiging) en hoeveel deze medewerker normaal gesproken doet.
2. Maak een lijst van bronnen die op de gewenste dag verschijnen waar vast een aantal medewerkers op staan. Sorteer deze per bron op prioriteit van de medewerker.
3. Verwijder de medewerkers die niet nodig zijn (omdat er al genoeg op een bron staan) uit de lijst van bronnen en voeg deze toe aan de lijst met Beschikbare Medewerkers. Maak meteen de (vaste) planning voor deze bronnen, zodat deze lijst niet weer hoeft worden bekeken.
4. Maak opnieuw een lijst van bronnen, deze keer alleen deze waar nog niemand voor was.
5. Breid de lijst uit met bronnen die van de vorige dag(en) overgebleven zijn.
6. Doorloop deze bronnenlijst en selecteer voor elke bron (indien mogelijk) een (willekeurige) medewerker uit de lijst met Beschikbare Medewerkers. Verwijder deze medewerker uit de lijst als er genoeg werk voor hem of haar is gevonden.
7. Als er nog medewerkers over zijn, geef aan dat ze geen bron hebben kunnen krijgen.
8. Als er nog bronnen over zijn, geef aan dat er geen medewerker gevonden kan worden voor de bron.

Het selecteren van een medewerker (stap 6) gaat als volgt:



De prioriteit ligt vast in het algoritme. Er wordt per bron geprobeerd deze zo volledig mogelijk uit te delen. Het gevolg is dat er slechts een overzichtelijk aantal bronnen overblijft. Een alternatieve aanpak waarbij van elke bron wel wat overblijft is niet gekozen. Het aantal medewerkers en bronnen is redelijk groot, en het gevolg zou dan zijn dat er vrij veel van hen

meerdere bronnen moeten doen, terwijl ze gewoon iets meer van een bepaalde bron hadden kunnen krijgen. Medewerkers die nog geen werk hebben gekregen worden willekeurig uit de lijst gekozen.

Als een deel van de planning al uitgevoerd is, en een aantal medewerkers al een bron heeft gekregen (uitgedeeld), is het niet mogelijk de planning voor dat deel te verwijderen. Bij het opnieuw maken van de planning wordt dan ook alleen dat deel verwijderd waarvan nog niets is uitgedeeld.

Bij het **bijwerken** van de planning worden alleen nieuwe bronnen en nieuwe medewerkers toegevoegd. Deze moeten dan handmatig worden ingedeeld. Als dit automatisch zou gebeuren zou het hele planningsalgoritme opnieuw doorlopen moeten worden en bestaat de mogelijkheid dat alle medewerkers ineens een andere bron toebedeeld krijgen. Aangezien deze situatie zich maar zelden voordoet, is het voor de gebruiker eenvoudig de nieuwe medewerker een bron toe te delen.

5.2 Algoritme voor uitdeling

Zoals gezegd is dit een taak voor de gebruiker van het programma, en is er dus geen algoritme ingebouwd. Wel wordt er bij elke stap van de gebruiker gecontroleerd of dit wel kan. Zo kan een gebruiker alleen maar delen van een bron aan een medewerker uitdelen als van deze medewerker geregistreerd is dat hij of zij die dag werkt. Ook wordt automatisch de (goede) datum van de bron ingevuld. Uiteraard zijn alleen die bronnen beschikbaar die er ook daadwerkelijk zijn.

5.3 Algoritme voor controle

Een van de belangrijkste doelen van de opdracht is het kunnen controleren of het uitgedeelde werk ook daadwerkelijk terug komt. Er wordt met veel verschillende medewerkers in het land gewerkt. Hierdoor komt er elke dag een groot aantal (verschillende) fouten voor. Deze fouten moeten zo goed mogelijk gedetecteerd worden.

5.3.1 Handmatige versie

Kort gezegd is er eigenlijk geen handmatige versie van de controle. Er wordt achteraf (als de vacatures al in de database zitten) gekeken naar vreemde data of aantallen. Dit zijn subjectieve beoordelingen van een controleur. In geval van tijdtekort lijdt de controle hier als eerste onder. In een aantal gevallen kan de oorsprong van gedetecteerde fouten worden herleid, maar dit is lang niet altijd het geval. De meer ervaren medewerkers met een goed geheugen kunnen afwijkingen detecteren, maar moeten dan alsnog zelf kijken of het een ernstige afwijking is of niet.

5.3.2 Geïmplementeerde versie

Het doel van de controle is een stevige basis leggen zodat alle significante afwijkingen worden gedetecteerd. Tevens is het van belang dat alle afwijkingen expliciet door een controleur worden goedgekeurd (dit in tegenstelling tot de passieve controle hierboven).

Het basisprincipe is gebaseerd op de aanname dat medewerkers proberen hun werk goed te doen. De hoogste prioriteit ligt bij het zoeken naar een zogenaamde "volledige overeenkomst" tussen wat is uitgedeeld en wat is ingeleverd. Om te kunnen onderscheiden wat goed en wat niet goed is, worden hier eerst de veel voorkomende situaties opgesomd:

- Het aantal vacatures klopt niet
- De datum van de bron klopt niet
- De naam van de bron klopt niet
- Het werk is nog niet binnen (of komt helemaal niet binnen)

Om alles nog ingewikkelder te maken, doen veel medewerkers het werk *gedeeltelijk* goed en gedeeltelijk verkeerd. Hiermee is ten dele rekening gehouden in de controle: het deel wat goed is kan automatisch gedetecteerd worden, zodat het deel wat niet goed is automatisch over blijft. Als een medewerker het echt heel erg bont maakt kan het systeem niet meer achterhalen wat er aan de hand is. Aangezien dan echter alles als fout wordt geclassificeerd, is de gebruiker snel in staat in te grijpen en heeft het systeem alsnog zijn werk gedaan: de controleur erop attenderen dat er iets heel erg fout is.

Voor de controle is een aparte tabel gereserveerd. In deze tabel wordt een verwijzing naar alle uitgedeelde klussen en al het ingeleverde werk opgeslagen. Op deze manier kan er niets overgeslagen worden, alles staat immers in de tabel.

Het algoritme dat gebruikt wordt voor de controle bestaat uit de volgende stappen:
1. Alle uit de Bestandsinfotabel verdwenen entries worden ook uit de Controletabel verwijderd (als er geen uitdeling bij hoorde) of op "Nog niet gedaan" gezet (als er wel een uitdeling bij hoorde).
2. Alle uitgedeelde klussen die nog niet in de tabel staan worden uit de Uitdelingstabel gelezen en toegevoegd met de status "Nog niet gedaan".
3. Alle binnengekomen porties die nog niet in de tabel staan worden op de volgende manier behandeld:
   - De medewerker die het werk gedaan heeft wordt opgezocht.
   - Alle entries in de Controletabel die bij de medewerker horen, waarbij de status niet op "OK" staat worden opgezocht.
   - De best passende entry wordt gekozen door op volgorde te kijken naar:
     1. bijna volledige match (bron en datum komen overeen, het aantal wijkt maximaal 5% af)
     2. Bron komt overeen, maar datum en aantal wijken af
     3. Datum komt overeen, aantal wijkt maximaal 5% af, bron wijkt ook af
     4. Datum komt overeen, bron en aantal wijken af
   - Als dit een entry oplevert, wordt deze geupdate met de bijbehorende status.
   - Als er geen passende entry wordt gevonden, wordt er een nieuwe entry in de tabel toegevoegd met als status "Niet uitgedeeld". In dit geval wijkt het binnengekomen werk zoveel af, dat het onmogelijk is te bepalen of het overeen komt met wat uitgedeeld is.

Het komt regelmatig voor dat medewerkers hun werk in twee delen opsturen. Het is niet de bedoeling dat de controle of het totaal klopt alsnog aan de gebruiker wordt overgelaten. Daarom is er na het bovenstaand algoritme nog een kleine verwerkingsslag nodig:
1. Bereken de totalen van alle bestanden die bij dezelfde entry uit de Uitdeling horen, waarvoor geldt dat er minimaal 2 bestanden zijn.
2. Doorloop de lijst van deze totalen en kijk of het totaal misschien wel binnen de (hierboven genoemde) grenzen van de getolereerde afwijking past. Zo ja, update dan de Controletabel zodat al deze entries nu de status "OK" krijgen.

In het statusveld kunnen meerdere fouten bijgehouden worden. De mogelijke inhoud van het statusveld is als volgt:

StatOK	0
StatOKGezet	1
StatAantal	2
StatDatum	4
StatBron	8
StatNotUitged	16
StatNotGedaan	32

De uitzondering is echter dat de status "OK" en "OKGezet" niet mogen voorkomen in combinatie met andere waarden.
In deze opzet is ervoor gekozen dat voor iedere entry in de Uitdeling een entry in de Controle wordt gecreëerd. Per teruggekomen portie wordt gezocht welke Uitdeling daarmee het beste overeenkomt of kan komen. De alternatieve aanpak (redeneren vanuit teruggekomen werk) is ook mogelijk. Er is echter voor de eerste methode gekozen omdat deze nauwer aansluit bij de werkelijkheid.

5.4 Overige algoritmes

De hier beschreven algoritmes zijn verwant aan de eerder genoemde algoritmes voor planning en controle. Ze behoren niet tot de kern van de applicatie, maar hebben een sterk ondersteunende functie.

5.4.1 Inlezen van vacaturebestanden

De informatie die uit vacaturebestanden wordt gehaald bestaat uit:

- Bestandsnaam
- MedewerkerID
- BronID
- Aantal vacatures
- Aantal vacatures uit snellijsten

Hiervoor worden de directories die als importdirectory (het BBS) zijn aangegeven doorlopen en wordt voor ieder bestand gekeken of het mogelijk een vacaturebestand is. Per bestand worden daarna de volgende stappen gedaan:

1. Als het bestand al eerder gezien was en niet is veranderd, wordt het overgeslagen.
2. Als het bestand is gewijzigd, wordt de oude informatie verwijderd en de nieuwe toegevoegd.
3. Als het bestand nieuw is, wordt de informatie opgeslagen in de Bestandsinfo tabel. De naam van het bestand wordt inclusief directory in hoofdletters opgeslagen.

5.4.2 Bepalen van weeknummers

In de applicatie komen net als in de organisatie twee verschillende weeknummeringen voor. Het eerste weeknummer is het officiële weeknummer, de week van het jaar. Het tweede nummer is altijd een week later, namelijk de week waarin de vacaturedatabase wordt opgeleverd. Het probleem met weeknummers is dat ze kunnen oplopen tot week 53 (terwijl er maar 52 weken in een jaar zitten). Ook kan het voorkomen dat week 1 dagen bevat die bij het vorige jaar horen.

Voor de bepaling van het weeknummer wordt gebruik gemaakt van de **ISO** (International Organization for Standardization) definitie van weeknummers. Dit komt neer op dezelfde nummers zoals deze in agenda's staan. ISO standaard **8601** zegt: "Weeknummer 1 van een jaar is de week waarin de eerste donderdag van dat jaar valt". Een alternatieve manier om dit te omschrijven is: "Weeknummer 1 is de week waarin 4 januari valt".

In het gebruikte algoritme wordt het weeknummer berekend van een willekeurige dag in een willekeurig jaar. Als het weeknummer van de huidige week nodig is, is dat dus de week met daarin de dag van vandaag. Wanneer het weeknummer van de database nodig is, wordt bij de al bekende datum (bijvoorbeeld die van vandaag) gewoon 7 dagen opgeteld, en wordt de daarmee verkregen datum aan het algoritme aangeboden.

Het algoritme ziet er als volgt uit:
1. Bepaal de datum waarop de eerste week van het gewenste jaar begint (zie onder).
2. Bereken het verschil in dagen tussen de gewenste dag en de dag waarop de eerste week begint (dag – dag_van_de_eerste_week).
3. Is dit verschil kleiner dan 0, doorloop dan dit algoritme met in plaats van het gewenste jaar, het gewenste jaar – 1.
4. Is dit verschil in weken (gedeeld door 7) groter dan 51, vraag dan de datum van de eerste week van het volgende jaar op en:
   - Geef als resultaat "1" als die datum kleiner of gelijk is aan de gewenste datum.
   - Anders: geef als resultaat "53".
5. Geef als resultaat: (verschil / 7) + 1, waarbij de deling naar beneden wordt afgerond.

In dit algoritme wordt meerdere keren verwezen naar het verkrijgen van de datum van de eerste week van het jaar, of liever, de datum van de eerste dag van de eerste week van het jaar. Dit kan dus een dag in het vorige jaar zijn. Hieronder volgt dat algoritme:
1. Bereken de datum van dag "04-01-jjjj" (waarbij "jjjj" het gewenste jaar is).
2. Als deze dag een zondag is, trek er dan 7 dagen af. Zo niet, trek er dan het dagnummer – 1 vanaf (dus bij maandag: 1, bij zaterdag: 6). Dit levert beide keren een zondag op.
3. De datum van de eerste dag van de week is het resultaat van stap (2).

Hierbij wordt gebruik gemaakt van de DayOfWeek() functie in Delphi, welke de dag van de week berekent (1=Zondag, 7=Zaterdag). Hierin is ook de 4e januari terug te vinden.

6 Het ontwerp van de applicatie

Het voorgaande is verwerkt in de uiteindelijke applicatie. Deze moet draaien op Windows (95/NT of hoger) computers binnen een netwerk. De applicatie is in **Delphi 5 Enterprise Edition** geschreven met behulp van de **Microsoft ADO (2.1) database** toegang tot een **Access database**. De keuze voor Delphi is gemaakt door de opdrachtgever. Delphi is redelijk eenvoudig te leren en het onderhoud aan de applicatie moet eventueel door een ander gedaan worden. De performance van Delphi is vrijwel gelijk aan die van C++ en levert dus geen problemen. Tevens heeft de externe partij die een nieuwe invoerapplicatie ontwikkelt aangegeven dat de taal ook de toekomstige Oracle database moet kunnen benaderen. De website van Borland laat weten dat Delphi hiervoor geschikt is.

De keuze voor ADO (2.1) is voor de hand liggend omdat deze componenten bij Windows 98 automatisch geïnstalleerd worden. Bovendien is ADO inmiddels een geaccepteerde standaard en heeft dus een vrij zekere toekomst. Na vele dagen uitproberen is overigens gebleken dat de "2.1" bij Windows 98 een "andere 2.1" is dan de versie "2.1" die te downloaden is bij Microsoft. De verschillen komen boven wanneer men bij (Access) databases probeert bepaalde queries uit te voeren (zoals "group by"). Aangezien dit probleem zich ook voordoet bij andere applicaties, is er inmiddels op het gehele netwerk een update gezet.

Er is voor Access gekozen omdat hierin redelijk snel databases te maken zijn. In de implementatie wordt niet of nauwelijks gebruik gemaakt van specifieke Access mogelijkheden. Dit maakt een latere conversie makkelijker.
Om het leertraject met betrekking tot Access te beperken, wordt alleen gebruik gemaakt van de mogelijkheid tabellen en queries op te slaan. Geavanceerde technieken vereisen goede (praktische) kennis van Access, wat in dit geval alleen maar tot een langdurig inleestraject zou leiden.

6.1 Functioneel ontwerp

Allereerst is er een functioneel ontwerp gemaakt van de applicatie. Hierin wordt gedeeltelijk beschreven wat de applicatie moet doen, en voornamelijk hoe alles er uit komt te zien.

6.1.1 Doel ontwerp

Dit ontwerp is bedoeld om te bepalen wat moet worden gerealiseerd. Achteraf kan worden vastgesteld of dit ook gebeurd is. In dit deel wordt de te realiseren Windows (95/98/NT) applicatie beschreven.
Een aantal onderdelen is tussen haakjes geplaatst. Deze onderdelen zijn in eerste instantie wel ontworpen, maar bleken later te veel extra tijd te kosten. Na overleg is besloten deze als optioneel te beschouwen.

6.1.2 Algemene functionaliteit

Voor ieder scherm of venster geldt dat er (afhankelijk van de situatie) een *Print*, *Sluiten* (of *OK*) en een *Annuleren* (of vergelijkbare) knop beschikbaar is. Deze functionaliteit spreekt voor zich en wordt dan ook niet bij elk venster opnieuw genoemd.
Bij een aantal acties, met name acties die gevolgen kunnen hebben voor het systeem, wordt om een bevestiging gevraagd. Hierbij is er elke keer een *OK* en *Annuleren* knop, of een *Yes* (*Ja*) en *No* (*Nee*) knop aanwezig.

6.1.3 Menustructuur

Hieronder wordt de menustructuur weergegeven die het programma zal krijgen. Over het algemeen zal hierbij de binnen het bedrijf gebruikelijke terminologie gehandhaafd worden. Uitzonderingen hierop worden alleen gemaakt als deze opzet conflicteert met standaardapplicaties.

Het *Hoofdmenu* (de balk in het hoofdscherm) krijgt de volgende structuur:

- Bestand
  - Instellingen
  - Opschonen
  - Einde
- Schema
  - Aanwezigheid
  - Planning
  - Uitdeling
  - BestandsInformatie
  - Controle
  - (Registratie Aantallen)
  - Registratie Import
  - (Loonadministratie)
- Onderhoud
  - Medewerker gegevens
  - Bron gegevens
  - (Stukloon gegevens)
  - (Uurloon gegevens)
  - Vestiging gegevens
- Help
  - (Help index)
  - Info

Het is prettig als dezelfde structuur ook op het hoofdscherm door middel van knoppen te benaderen is. Hiervoor kan een zogenaamde "buttonbar" gebruikt worden met een icoontje van de meest gebruikte vensters.

6.1.4 Onderdelen van het menu

Hieronder worden de verschillende onderdelen van het menu besproken. Per menu-item (functie) wordt aangegeven hoe het bijbehorende venster of scherm er uit komt te zien.

6.1.4.1 Menu: Bestand

Het algemene menu met daarin de gebruikelijke functies voor een programma.

Functie: Instellingen

De standaardinstellingen kunnen hier gewijzigd worden. In eerste instantie zullen alleen de volgende instellingen aanwezig zijn:

- Database directory
- Bronnenlijst directory
- Import directories

Al deze instellingen kunnen met de knop *Wijzig* gewijzigd worden. Voor elke instelling is een *Herstel standaard* knop aanwezig welke de oorspronkelijke instellingen herstelt.

Functie: Opschonen
Elke maandag wordt automatisch de opschoning van de database aangeroepen. Met deze functie kan dit handmatig worden gedaan. De mogelijkheden in het scherm zijn beperkt tot het schoonmaken (*Schoonmaken (verwijderen) van oude gegevens*) en het annuleren (*Sluiten (zonder eerst op te schonen)*) van het scherm.

Functie: Einde
Beëindigt het programma.

6.1.4.2 Menu: Schema

In het menu Schema is het mogelijk de schema's op te roepen die nodig zijn voor de personeelsplanning, de registratie en de controle. Iedere functie geeft een nieuw venster weer met daarop de voor dat schema relevante informatie en functionaliteit.

Functie: Aanwezigheid
Het scherm toont de namen van alle medewerkers en hun beschikbaarheid in de huidige, komende (en eventueel vorige) weken. Door middel van selectie is het mogelijk een medewerker voor een dag af te melden (*Afmelden*), of weer opnieuw aan te melden (*Aanmelden*). In de lijst is opgenomen of een medewerker aanwezig is "**o**" of niet "**X**".

Permanent (of langdurig) aan- of afmelden voor een bepaalde dag in de week is hier niet mogelijk, dit moet veranderd worden in de personeelsgegevens.

Functie: Planning (overzicht)
De planning is de tweede en een van de belangrijkste stappen in het automatiseringsproces. Het bestaat uit een scherm met daarin alle relevante gegevens voor het plannen van personeel voor de gekozen dag. Het is mogelijk met de knoppen *Vorige* en *Volgende* naar andere dagen te kijken. Tevens zijn hier knoppen aanwezig met *Vandaag* en *Zaterdag* om snel deze (belangrijke) dagen te kunnen bekijken.

Het scherm bevat standaard een lijst met alle bronnen die gedaan worden met daarachter de medewerkers die ingedeeld zijn met een aantal vacatures. Het is ook mogelijk de volgorde om te draaien: een lijst van alle medewerkers met de bijbehorende bron. Het zoeken op een bron of medewerker is rechtstreeks te doen door middel van de *Zoek* knop.

Wijzigingen zijn door te voeren door op *Aanpassen* te klikken.
Door middel van kleuren is te zien wat de status van een bron is. Als er niemand voor is, wordt de tekst in **rood** afgebeeld. In het andere geval is de tekst gewoon zwart.
Handmatige wijzigingen worden in **blauw** weergegeven. Als een planning veilig is, wordt deze in **groen** weergegeven. Om ook op een afdruk op papier (zonder kleurenprinter) duidelijk te kunnen zien wat er aan de hand is, wordt voor alle kleuren ook een apart teken gebruikt. De legenda van gebruikte kleuren en tekens staat altijd onderaan de pagina.

Met de knop *Waarschuwingen* zijn alle waarschuwingen op een rij te zien.

De planning is (opnieuw) te maken met de knop *Planning maken*. Deze laat een nieuw scherm zien met daarop de mogelijkheden om de huidige planning te *Verwijderen*, *(Opnieuw) maken*, of *Bijwerken*.

Functie: Planning (aanpassen)
De layout van het scherm wijzigt als er voor *Aanpassen* gekozen wordt. Hier is het mogelijk met *Overzicht* terug te gaan naar het overzicht. Verder kunnen de volgende wijzigingen gemaakt worden:

- Een bron wordt bij een medewerker weggehaald
- Een bron wordt (gedeeltelijk) aan een medewerker toebedeeld

Dit kan gedaan worden door de bron van het ene naar het andere venster te slepen, of de knoppen met pijltjes te gebruiken.

De gewijzigde planning kan vervolgens opgeslagen worden met *Opslaan* of genegeerd worden met *Annuleren*.

Functie: Uitdeling
Geeft ongeveer hetzelfde scherm als het aanpassen bij de planning. Hierbij is het echter niet mogelijk de indeling (verdeling, planning) aan te passen. Het enige wat hier kan worden toegevoegd of gewijzigd is de hoeveelheid die uitgedeeld is. Dit kan gedaan worden door de naam van een bron te selecteren en uit te delen.

In de aanwezige lijsten worden de namen van medewerkers weergegeven, de bronnen die ze hebben gekregen, de bronnen die in de planning aan hen toebedeeld zijn, en de bronnen die nog over zijn.

Bij het *Uitdelen van geplande bron* wordt nog de mogelijkheid gegeven het geplande aantal vacatures aan te passen aan de werkelijkheid.

Functie: BestandsInformatie
De bestandsinformatie geeft een venster weer met daarop de namen van medewerkers in een zogenaamde boom. Per medewerker is er te zien welke bestanden deze heeft gestuurd, en hoeveel vacatures er in het bestand staan. Het aantal vacatures is per bron en datum zichtbaar.

Met de knop *Vernieuwen* worden de nieuw binnengekomen bestanden ingelezen en opgenomen in de lijst (boom).

Functie: Controle
Het controlescherm geeft net als de uitdeling een lijst met de namen van medewerkers. In dit geval worden alleen die medewerkers getoond die werk hebben gekregen en/of ingeleverd. Bij selectie van een medewerker wordt het uitgedeelde werk boven, en het ingeleverde werk onder weergegeven. Helemaal onder verschijnt bij selectie van een portie de status van de selectie.

De gebruiker kan een als fout geclassificeerde regel (combinatie van Bron en Datum) goedkeuren met de knop *Geef goedkeuring*. Dit kan niet ongedaan gemaakt worden.

Met de knop *Vernieuwen* worden de nieuw binnengekomen bestanden ingelezen en opgenomen in de lijst en meteen de status per portie bepaald.

(Functie: Registratie aantallen)
Het registratiescherm is afgeleid van de papieren versie die op dit moment gebruikt wordt. Het scherm bestaat uit een lijst met alle namen van de medewerkers die de huidige week werken. Ook hier is het weer mogelijk met *Vorige* en *Volgende* andere weken te bekijken.

De lijst is gesplitst in twee delen. Het eerste deel bevat de medewerkers die al werk ingeleverd hebben. Deze lijst kan gesorteerd worden op naam van de medewerker of op volgorde van binnenkomst. Het tweede deel bevat de namen van medewerkers van wie het werk nog niet binnen is.

Door op een naam te dubbelklikken wordt de mogelijkheid gegeven de verdeling van de medewerker in te geven. Hier is ook de mogelijkheid aan te geven dat de medewerker klaar is en dus niet meer zal inleveren. Als een medewerker op uurbasis werkt, is hierbij het aantal uren op te geven.

Er wordt automatisch elke minuut gekeken of er een medewerker werk heeft ingeleverd. De knop *Vernieuwen* zorgt ervoor dat de informatie onmiddellijk wordt ververst.

Functie: Registratie Import
Bij de importregistratie zijn twee kolommen zichtbaar met bestandsnamen. Links staan de bestanden die nog geïmporteerd moeten worden. Rechts staan de bestanden die al geïmporteerd zijn. Door middel van dubbelklikken of verplaatsen kan een bestand als *geïmporteerd* worden gemarkeerd.

(Functie: Loonadministratie)
De lonen worden per maand uitbetaald. De grens van een maand ligt altijd op een zaterdag. Met de knop *Selecteer weken* zijn de weken te selecteren die bij de huidige maand horen. Vervolgens kunnen met de knop *Bereken* de lonen berekend worden.

6.1.4.3 Menu: Onderhoud
In het onderhoudsmenu zijn de standaardtabellen (stamtabellen) aan te passen.

Functie: Werknemer gegevens
Alle relevante gegevens van de werknemer zijn hier oproepbaar. De stamgegevens kunnen veranderd worden, bijvoorbeeld bij verhuizing of verandering van beschikbaarheid met de knop *Wijzigen*.

Om niet alle namen in een keer te hoeven bekijken is het mogelijk een aantal (actieve of niet actieve) medewerkers te selecteren

Functie: Bron gegevens
De bronnenlijst wordt overgenomen uit het controleprogramma. Met de knop *Wijzigen* kunnen de stamgegevens van een bron gewijzigd worden. In een nieuw venster kunnen dan de (te veranderen) gegevens van de bron aangepast worden en vervolgens worden opgeslagen.

(Functie: Stukloon gegevens)
In het scherm kunnen de bruto betalingen voor de verdeling worden aangepast. Met *Nieuw* kan er een nieuwe categorie worden toegevoegd (buiten de standaard groot, middel en klein). Gegevens kunnen met *Opslaan* worden opgeslagen en met *Verwijder* worden verwijderd.

(Functie: Uurloon gegevens)
In het scherm kunnen standaard bruto uurlonen worden ingevuld en met *Opslaan* worden opgeslagen, of met *Verwijder* worden verwijderd.

Functie: Vestiging gegevens
Hier staan de namen (en nummers) van de vestigingen. Dit zijn de kantoren waarop gewerkt wordt. Een vestiging (anders dan de al aanwezige) kan worden toegevoegd (*Toevoegen*), of van naam worden gewijzigd (*Wijzigen*). Met *Verwijder* kan een niet meer bestaande vestiging worden verwijderd.

6.1.4.4 Menu: Help
In dit menu zijn de helpindex en informatie over het programma op te vragen.

(Functie: Helpindex)
Toont het helpscherm met informatie voor bediening van het programma.

Functie: Info
Toont de versie- en productinformatie.

6.1.5 Opmerkingen
In de implementatie kan op sommige punten afgeweken worden van dit functioneel ontwerp. Het zal hier voornamelijk kleine aanpassingen in de Grafische Interface betreffen die het werken met de applicatie eenvoudiger of prettiger moeten maken.

6.2 Implementatietraject

In het functioneel ontwerp is in grote lijnen de implementatie terug te vinden. Allereerst is de planning opgezet. Vanuit een in eenvoudige vorm werkende planning is de uitdeling tot stand gekomen. Daarna zijn alle zijtrajecten (verzamelen van gegevens en dergelijke) geïmplementeerd. Als laatste is de controle ontworpen die alle gewenste gegevens kan combineren.

Tijdens de implementatie is af en toe van de oorspronkelijke specificatie afgeweken, de redenen hiervoor worden per afwijking aangegeven.

6.2.1 Doel van de opdracht
De kern van het doel van de opdracht is een betere waarborging van de kwaliteit. In dit geval wordt met kwaliteit bedoeld dat het resultaat van de invoer van vacatures overeenkomt met de verdeling van die vacatures op kantoor. De inhoudelijke kwaliteit van het werk staat hier dus buiten. Voor de applicatie geldt dus dat iedere (significante) afwijking gedetecteerd moet worden.

6.2.2 Algemene technieken
In het vorige hoofdstuk (over de database) is al een aantal keren vermeld dat de implementatie van de Access database niet altijd even vlekkeloos is verlopen. Bij elk onderdeel van de applicatie zijn de gevolgen hiervan terug te vinden. Dit heeft geleid tot het maken van keuzes met betrekking tot de gebruikte technieken.

In de eerste plaats is er het probleem met de directe toegang tot de actuele gegevens in de database. Het overijverige caching systeem levert niet altijd de net gewijzigde informatie, om deze informatie wel te krijgen moet de connectie met de database opnieuw geopend worden, hetgeen veel tijd kost. Wanneer een kleine wijziging zoveel tijd kost, zorgt dit voor irritatie bij de gebruiker. Het alternatief is een lijst (lees: tabel) in het geheugen laden, en de bewerkingen op de gegevens pas later op te slaan.

Aangezien dit een vrij ingrijpende verandering is ten opzichte van een rechtstreekse benadering, wordt hieronder een overzicht gegeven van de voor en nadelen van beide methodes:

Directe benadering van Queries en Tabellen:

Voordelen	Nadelen
Met SQL is het vrij eenvoudig de gewenste gegevens te laten zien.	Er kunnen geen extra (tijdelijke) gegevens gebruikt worden, omdat deze tot wijzigingen in de database zouden leiden.
Met gekoppelde tabellen en Queries is het vrij eenvoudig ook de gewenste gekoppelde resultaten te laten zien die bij bepaalde gegevens horen.	Als een soort koppeling vaker nodig is, zal deze elke keer weer opnieuw gemaakt moeten worden (of in de visuele omgeving, of in code). Alleen het principe is opnieuw te gebruiken (waardoor alleen het maken zelf een tweede keer sneller gaat).
Er zijn visuele databasecomponenten die speciaal bedoeld zijn voor de weergave van tabellen en/of Queries.	De visuele databasecomponenten zijn vrij beperkt en kunnen alleen met voorgeprogrammeerde situaties omgaan.
In de al gemaakte experimentele versie is het grootste gedeelte al geïmplementeerd.	Het is lastig andere gegevens dan die van het huidige record te bekijken (op te vragen) of aan te passen. Dit heeft ook tot gevolg dat de Drag-en-Drop lastiger wordt.
	In de al gemaakte versie blijkt het zeer lastig te zijn om de dataset te veranderen (telkens "current record errors").

Implementatie met interne lijsten:

Voordelen	Nadelen
In de interne representatiestructuur is het mogelijk tijdelijke informatie op te slaan. Deze kan gewijzigd worden, zonder dat dit in de database ook wijzigingen teweeg brengt.	Koppelingen tussen gegevens en resultaten (verschillende tabellen) moeten zelf gemaakt worden (door middel van het afvangen van muisklikken).
De procedures en lijsten met betrekking tot het ADT zijn generiek te maken, waardoor ze waarschijnlijk vaker te gebruiken zijn.	Er moet een ADT gemaakt worden (object met gegevens) dat in een lijst- of boomstructuur gebruikt kan worden. Verder moeten er operaties op deze lijsten mogelijk zijn.
Bij wijzigingen (op het scherm) hoeft er nog niks in de database geschreven te worden.	Wijzigingen moeten later zelf in de database worden geschreven.
Gegevens kunnen in de gehele lijst opgevraagd en gewijzigd worden, dus niet beperkt tot het huidige 'record'.	
De Drag-en-Drop is met een lijst heel makkelijk (bijna zonder code) te implementeren.	

De beide mogelijkheden lijken elkaar niet zoveel te ontlopen. Beide keren is er een hoeveelheid extra werk nodig om het geheel werkend te krijgen. Het is echter een illusie dat er dus weinig verschil is en de directe aanpak de meest logische is.

De volgende punten hebben zwaar meegeteld in de afweging:

- Drag-en-Drop werkt niet of niet zoals verwacht bij de database componenten.
- Foutmeldingen die een onbekende oorsprong hebben, of voortkomen uit de omgang met huidige records in de database, zijn zeer lastig te voorkomen. Deze mogen tijdens dagelijks gebruik niet voorkomen.
- De applicatie mag niet (te) traag worden.

Het werk, en de onzekerheid die het oplossen van deze punten met zich meebrengt, weegt op tegen de hoeveelheid extra werk die voor interne lijsten gedaan moet worden.
Tijdens de implementatie is waar mogelijk gebruik gemaakt van de directe benadering van de database. Waar dit problemen zou kunnen opleveren is gekozen voor het bijhouden van de informatie in het geheugen. De keuze is dus gevallen op de oplossing met de minste of minst ernstige nadelen, bekeken per situatie.

Sommige database foutmeldingen blijken niet meer voor te komen als de applicatie niet in een debugomgeving wordt getest. De oorzaak hiervan is tot op heden onbekend, maar omdat de applicatie in elke situatie moet werken, is dus gekozen voor de veiligere aanpak.

Ten tweede is er nog een standaardprobleem met betrekking tot het afdrukken van informatie op een printer. Het is zeer eenvoudig de inhoud van een tekstveld af te drukken. Hiervoor is alleen het opgeven van de printerspecificatie met betrekking tot papiergrootte nodig. Iets anders dan tekst afdrukken is echter vrijwel onmogelijk, ook al bevat elke component (tabel, lijst, enz.) een methode hiervoor. Het resultaat is dan meestal een microscopisch kleine afbeelding (1x1 centimeter). Er zijn andere technieken (report-printers) die hiervoor gebruikt kunnen worden. Het nadeel is dan dat de layout twee keer op verschillende manieren geprogrammeerd moet worden en bij latere wijzigingen, dit wederom moet gebeuren. De keuze is daarom gevallen op een gratis te verkrijgen component die de inhoud van een component (tabel) redelijk natuurgetrouw naar de printer kan sturen.

6.2.3 Planning

De implementatie van het planningsgedeelte is niet het belangrijkste onderdeel. Het is echter wel de basis voor de rest van het systeem en neemt dus een belangrijke plaats in het geheel in. Er is gezorgd voor een systeem met een zo stevig mogelijke basis, een automatische indeling die daarna met de hand kan worden aangepast.

Oorspronkelijk was in het ontwerp een onderscheid opgenomen tussen het plannen en het uitdelen van een hoeveelheid werk. De reden voor dit onderscheid kwam voort uit het feit dat in de oude situatie het plannen op vrijdag gebeurde en het uitdelen door andere medewerkers op zaterdag. Op het moment van uitdelen was de planning niet meer dan een indicatie, waar adhoc vanaf geweken werd omdat dit makkelijker leek.

Nadat het planningsalgoritme in een gevorderd stadium was gekomen, rees de vraag of dit onderscheid wel echt nodig was. Omdat de geplande situatie twijfel veroorzaakte, zijn de voor- en nadelen van de twee verschillende aanpakken hieronder op een rij gezet.

Onderscheid maken tussen planning en uitdeling:

Voordelen	Nadelen
Bij het uitdelen is de planning beschikbaar en kunnen dezelfde controles gebruikt worden als bij het handmatig aanpassen van een planning.	Bij het uitdelen moet er goed rekening worden gehouden met het feit dat een uitdeling een ander deel van de planning onmogelijk kan maken. Extra controles zijn dus nodig.
De oorspronkelijk gemaakte planning wordt vanaf een bepaald moment niet meer gewijzigd, en is dus later terug te vinden.	Het vergt meer discipline om niet zomaar even de hele planning om te gooien.
In de database is de opslag van planning en uitdeling gescheiden en zijn verschillen (ook achteraf) nog na te gaan.	Er moet een duidelijke afspraak zijn over wanneer een planning moet worden veranderd en wanneer dat in de uitdeling moet.
Bij het uitdelen is minder vereist dan bij het plannen. De hoeveelheid werk mag bijvoorbeeld behoorlijk afwijken omdat dit de actuele situatie is.	Een gesplitst systeem kan te veel mogelijkheden (toeters en bellen) geven, en zou onduidelijkheden in de hand kunnen werken.
De laagstructuur in de database kan letterlijk worden overgenomen.	Het scherm waarin de wijzigingen in de planning gemaakt worden, moet gedeeltelijk overgenomen worden voor de uitdeling.

De keuze is gevallen op het scheiden van planning en uitdeling. Hierbij hebben de volgende factoren een grote invloed gehad:

- Overzichtelijkheid en aansluiting bij de huidige (oude) situatie.
- Het behoud van de laagstructuur in de database.

Een vergelijkbaar geval deed zich voor bij het bepalen van een planning (verdeling) van het werk. In eerste instantie was het de bedoeling om twee tabellen te gebruiken, de eerste met *Standaardaantallen* en de tweede met een *Vasteverdeling*. In de eerste tabel zouden alleen aantallen komen die een medewerker standaard aan kan op een bepaalde dag. De tweede tabel zou de gegevens moeten bevatten met de vast afgesproken aantallen die een medewerker uit een afgesproken bron doet. Uiteindelijk is besloten deze twee tabellen samen te voegen tot slechts één, waarbij dan bepaalde waardes leeg gelaten mogen worden. De redenen voor deze keuze zijn:

- Oorspronkelijk was het verschil tussen de tabellen gemaakt om een strikte scheiding te hanteren tussen een aantal en een *vast afgesproken* aantal en bron. Het planningsalgoritme moest hierdoor gegevens uit meerdere tabellen halen en telkens vergelijken. Door alles in dezelfde tabel te zetten wordt snel duidelijk waar in de planning de prioriteiten liggen.
- Door een wijziging in het uitgangspunt voor de planning is het nu eigenlijk noodzakelijk van elke medewerker zijn of haar standaard aantal te weten. Bovendien moet een voorkeursbron kunnen worden opgegeven. Hiervoor was een uitbreiding met een paar velden noodzakelijk. Omdat de tabellen hierdoor nog meer op elkaar gingen lijken was een combinatie de meest logische keuze.

Bovendien sluit het gebruik van één tabel beter aan bij de intuïtieve weergave bij medewerkergegevens. Voor de gebruiker is het eenvoudiger een aantal toe te kennen en dan achteraf te beslissen of dit aantal geschat is, of tot een vaste verdeling behoort. Tijdens een

kort interview met twee van de controleurs bleek zelfs dat het verschil tussen een "vaste" en een "standaard" verdeling hun niet duidelijk was. Met slechts één tabel en één weergave daarvan werd het verhaal voor de controleurs duidelijker. Intuïtief begrepen ze ook direct de eventuele toevoeging van het woord "vast" bij een al gemaakte verdeling.

Tenslotte is er nog sprake van een wijziging met betrekking tot de indeling van de week. In de oorspronkelijke planning was het de bedoeling een zogenaamde Weekplanning en een aparte Zaterdagplanning op te nemen. Dit moest het gebruik eenvoudiger maken. Het grootste deel van het werk werd op zaterdag gedaan, en door de week werd adhoc met de op dat moment geldende situatie omgegaan. Gedurende het traject van de implementatie is echter een grotere behoefte ontstaan aan een gestructureerde planning voor de gehele week. In overleg is toen besloten een algemene *dagelijkse* planning mogelijk te maken, gelijk aan de al geïmplementeerde planning voor zaterdag.

Het voor de planning gebruikte algoritme is in het vorige hoofdstuk al uitvoerig beschreven. De weergave op het scherm is op twee manieren geïmplementeerd: een overzichtsscherm en een wijzigingsscherm. Het overzichtsscherm komt grotendeels overeen met een planning op papier, alle bronnen en medewerkers staan onder elkaar met daarbij de aantallen.
Het wijzigingsscherm biedt de mogelijkheid per medewerker een bron toe te voegen of te verwijderen. Bij het toevoegen kan dan gekozen worden voor de hoeveelheid. Wijzigingen kunnen gemaakt worden door bronnen van het ene naar het andere venster te slepen, of door de aanwezige knoppen te gebruiken.

6.2.4 Uitdeling

Zoals hierboven al genoemd, wordt de uitdeling als een apart onderdeel beschouwd. De implementatie (weergave) heeft echter wel een vergelijkbare basis als het wijzigingsscherm van de planning. Bij het uitdelen is geen overzichtsscherm, dit is ook niet nodig omdat pas bij de controle gekeken moet worden of alles klopt.
Het scherm geeft de gebruikelijke indeling met links de medewerkers en rechts vensters met bronnen die al dan niet zijn uitgedeeld.

6.2.5 Controle

De controle is in drie delen gesplitst, het belangrijkste deel waarin de kwaliteit (correctheid) wordt gecontroleerd, wordt ondersteund door informatie over binnengekomen bestanden en verwerkte (geïmporteerde) bestanden.

6.2.5.1 Bestandsinformatie

Om de bestandsinformatie effectief weer te geven is gekozen voor een boomweergave waarbij de namen van medewerkers op het hoogste niveau zichtbaar zijn. Per medewerker wordt de mogelijkheid geboden te kijken welke bestanden deze medewerker heeft opgestuurd. Per bestand wordt dan per bron weergegeven hoeveel vacatures ervan gedaan zijn.

De informatie die nodig is om dit alles weer te geven wordt direct uit de tabel *Bestandsinfo* gelezen waarin de namen van alle binnengekomen bestanden staan en hoeveel vacatures er van een bron per medewerker in dat bestand staan.

6.2.5.2 Correctheidscontrole

Het belangrijkste onderdeel van de controle bestaat uit het controleren of het werk dat uitgedeeld is ook correct is gedaan. Een registratie op papier van het uitgedeelde werk, is bij terugkomst moeilijk terug te vinden.

In het oorspronkelijke Functioneel Ontwerp is sprake geweest van een aantal acties die uitgevoerd kunnen worden door de gebruiker. Deze acties zijn alleen nodig als het programma niet automatisch zelf een goedkeuring kan geven. Het ging in dit geval om de volgende mogelijkheden:
1. Goedkeuring geven
2. Fout negeren
3. Fout bevestigen

Het doel van de applicatie is het signaleren van fouten en de gebruiker de mogelijkheid geven in te grijpen. Dat ingrijpen kan in de praktijk op meerdere manieren (hierbij moet gedacht worden aan het bellen van de medewerker, het verwijderen van bestanden, of bijvoorbeeld het uitstellen van de verwerking). Al deze acties zijn achteraf gezien samen te vatten tot twee effectieve mogelijkheden:
1. Goedkeuring geven bij niet relevante afwijkingen
2. De fout bevestigen en na eventuele ondernomen actie verder gaan

Om het systeem niet te ingewikkeld te maken, is er dus voor gekozen slechts twee mogelijkheden daadwerkelijk te implementeren. Het programma zal net zo lang de gevonden fout presenteren totdat de gebruiker deze zijn of haar goedkeuring geeft.
Het gevolg is dat gevonden fouten genegeerd mogen worden als de gebruiker weet dat er nog aan gewerkt wordt (nummer 2. van het oorspronkelijke ontwerp). Tenslotte kan de gebruiker goedkeuring geven met betrekking tot de huidige situatie (nummer 1. en 3. van het oorspronkelijke ontwerp). Dit geheel is voldoende om de kwaliteit te kunnen garanderen.

Het voor de controle gebruikte algoritme is in het vorige hoofdstuk al uitvoerig beschreven. De weergave op het scherm is eenvoudig gehouden: een lijst met medewerkers links. Bij het aanwijzen van een medewerker wordt rechts de informatie getoond waarin staat wat deze medewerker heeft gekregen en wat hij of zij heeft ingeleverd. Onderaan staat uitgebreide informatie over de correctheid en de mogelijkheid goedkeuring te geven.

6.2.5.3 Registratie import
In de applicatie is de registratie van de import beperkt tot een scherm dat een lijst van nog niet geïmporteerde en een lijst met al wel geïmporteerde bestanden weergeeft. Het is de taak van de gebruiker om op het moment van importeren een bestand ook als zodanig te markeren. De lijst van bestanden is gesorteerd op datum van binnenkomst, dit levert een "first in, first out" rij op. Het is niet verplicht deze volgorde te handhaven bij het importeren.

Op het moment is het onmogelijk na te gaan welke bestanden zijn geïmporteerd en welke niet, het vacature controleprogramma geeft hier geen enkele betrouwbare indicatie voor. Als dit in de toekomst verandert, is het eenvoudig de registratie aan te passen en er een automatische importcontrole van te maken.

6.2.6 Onderhoud
De term *Onderhoud* moet gezien worden als het "onderhoud van stamgegevens". In de database zijn dit dus de Medewerkers, Bronnen, Vestigingen en standaard (of vaste) verdeling.

Medewerker- en brongegevens moeten worden overgenomen uit een bestaande (FoxPro) database. Om dit technisch mogelijk te maken, moeten een aantal extra's op de pc

geïnstalleerd worden. De applicatie kan echter op iedere pc werken, maar kan dan niet overal de externe databases benaderen. Een bijkomstigheid is het feit dat het werken met deze externe databases nogal foutgevoelig is. Dit alles heeft geleid tot een kleine extra applicatie die de overdracht van gegevens mogelijk maakt. Binnen het systeem is dus alleen manipulatie van de eigen gegevens mogelijk (een integratie van de extra applicatie in het ontworpen systeem is in een later stadium wel mogelijk).

6.2.6.1 Medewerkers

Het overzicht van de medewerkergegevens is gebaseerd op een standaard Master-Detail scherm met voornamelijk informatie uit de tabel *Persoonsgegevens*. In een lijst staan de belangrijkste gegevens van medewerkers (code, naam en adres). Wanneer een medewerker geselecteerd wordt, worden de overige bekende gegevens van deze medewerker boven in het scherm weergegeven. In het overzicht is het niet mogelijk gegevens te wijzigen, dit om per ongeluk gemaakte wijzigingen te voorkomen.

Bij het wijzigen van de gegevens (knop *Wijzig gegevens*) wordt een nieuw scherm getoond met daarop dezelfde gegevens, maar dan zonder de lijst. Hierin wordt de mogelijkheid gegeven onderhoud te plegen met betrekking tot deze gegevens. Tevens is het hier mogelijk de standaard (vaste) verdeling voor de geselecteerde medewerker aan te passen. Deze verdeling wordt in de database in de tabel *Vasteverdeling* opgeslagen.

6.2.6.2 Bronnen

De gegevens van de gebruikte bronnen is in de externe database beperkt tot de naam en een uniek nummer. In de eigen database kan veel meer informatie worden opgeslagen. De lijst van gegevens die getoond wordt bij bronnenonderhoud is een complete lijst van alles wat van de bron bekend is en in de tabel *Bronnenlijst* staat. Van een geselecteerde bron kunnen bepaalde gegevens (zoals verschijning) gewijzigd worden. Hiervoor is dan een apart scherm beschikbaar.

Als hulpmiddel worden de gegevens uit de tabel *Vasteverdeling* ook apart weergegeven, zodra een bron geselecteerd is waar standaard een of meer medewerkers op staan.

6.2.6.3 Vestigingen

Het onderhoud aan de vestigingen is alleen voor de toekomst belangrijk. Het aantal vestigingen zal niet snel gewijzigd worden. De mogelijkheid om nieuwe vestigingen toe te voegen of al bestaande te wijzigen is in het programma opgenomen door middel van een eenvoudig scherm met daarin de namen van de vestigingen en bijbehorende toevoeg- en verwijderknoppen.

7 Verdere ontwikkelmogelijkheden

In de oorspronkelijke opdracht werd gesproken over een systeem dat eigenlijk alles zelf zou moeten kunnen. Er is meteen aan het begin ook afgesproken dat een aantal zaken hogere prioriteit hadden dan andere. De wens van de opdrachtgever is in de loop van het project meer gespecificeerd. Er is meer nadruk gelegd op de planning en controle en minder op de administratie. De redenen hiervoor liggen voornamelijk in een aantal incidenten die voorkomen hadden kunnen worden als de planning en controle minder adhoc door controleurs gedaan zou worden. Een programma wat bepaalde problemen automatisch signaleert heeft daardoor een hogere prioriteit gekregen.

Na overleg is besloten het administratieve gedeelte achteraf (inclusief loonberekening) buiten de opdracht te laten. In de ontwikkeling van het programma en de database is vanaf het begin wel rekening gehouden met deze delen. In een later staduim kunnen deze alsnog geïmplementeerd worden.

Van de registratie is wel het deel dat de aantallen vacatures bijhoudt opgenomen. De controle is gebaseerd op deze aantallen. Er moet immers gecontroleerd worden of het uitgedeelde deel ook daadwerkelijk is teruggekomen, omdat dit de belangrijkste bron is voor de vulling van de vacaturedatabase. De controle van de verdeling van de thuiswerker vindt achteraf plaats en is met de aantallenoverzichten al minder foutgevoelig geworden.

Er is een selectie gemaakt met betrekking tot de gegevens die per medewerker bewaard kunnen worden. De informatie die samenhangt met loonbelasting, inschrijving bij het GAK en de daarbij behorende data is in het programma niet opgenomen. De database en de schermen bevatten wel de benodigde velden, maar er kunnen geen waarden worden ingevuld.

Aangezien de applicatie volledig afhankelijk is van de gegevens die in het bestaande Controleprogramma staan (medewerkers- en bronnenlijsten), is er voor de veiligheid een externe applicatie ontwikkeld die deze gegevens kan overzetten. De bestaande database mag in geen geval gewijzigd worden. Een update van de gegevens in PCC moet dus als onderhoud gezien worden. Een veilig (automatisch) update-algoritme kan later in de applicatie toegevoegd worden.

7.1 Aanbevelingen

Op basis van de ervaringen met het programma en de ontwikkeling daarvan zijn er een aantal aanbevelingen te doen. Hieronder volgen een aantal mogelijke uitbreidingen en wijzigingen.

1. De Access database omzetten naar Oracle. Er is binnen het bedrijf nu een Oracle server en deze kan veel sneller de gegevens leveren. De combinatie Access-ADO is blijkbaar nog niet perfect en werkt nogal traag.
2. Binnen de database constraints toevoegen. In combinatie met het hierboven genoemde punt zal dit de integriteit van de database waarborgen. Dit wordt al binnen het programma gedaan, maar in Oracle is dat nog veiliger te realiseren.
3. De database van medewerkers wordt nu overgezet en aangepast binnen het programma. Een vergelijkbare tabel bestaat ook binnen Oracle en zou met een koppeling gebruikt kunnen worden om automatisch de nieuwste gegevens van medewerkers te bewaren.
4. In de controle niet meer door de gebruiker laten aangeven wat geïmporteerd is, maar deze gegevens opvragen. Dit is een mogelijkheid als in het nog nieuw te maken

controleprogramma voor de vacatures bij de import een systeem gebruikt wordt wat door dit programma dan gelezen kan worden.
5. Toevoegen van registratie van aantallen (klein, middel, groot) die de medewerkers doorgeven, en de daarbij behorende loonberekening. Eventueel met het direct exporteren naar een voor de boekhouder geschikt formaat.
6. Bij de waarschuwingen in de planning het aantal en een groepering toevoegen. Zo zouden de waarschuwingen een bepaalde prioriteit kunnen krijgen en is er per planning terug te vinden in hoeverre dit een veilige situatie is.
7. Het planningsalgoritme meer dynamisch maken. Nu wordt er een aantal regels gevolgd die ten tijde van het ontwerp het beste waren. Zonder al te veel moeite is het mogelijk een aantal scenario's te bedenken die een andere planning opleveren met prioriteiten die dan door de gebruiker zijn op te geven.
8. Telefoonnummers scheiden: één hoofdnummer dat bij een medewerker (verplicht) wordt opgenomen, en overige nummers in de aanwezige telefoonlijst. Dit maakt het ook eenvoudiger een telefoonlijst af te drukken met het geprefereerde telefoonnummer.

7.2 Overige opties

Gedurende de ontwikkeling zijn een aantal extra handigheden bedacht die de interface en het gebruik eenvoudiger kunnen maken. Dit zijn over het algemeen geen functionele veranderingen, maar extra mogelijkheden voor de gebruiker.

1. Gebruik maken van de rechter muisknop. In een aantal schermen is het zeer goed mogelijk voor de situatie relevante informatie onder de rechter muisknop te verbergen. Hierbij moet gedacht worden aan het aantal vacatures of bronnen, als op de naam van een medewerker in de planning gedrukt wordt. Of een standaardaantal of gebruikelijke werkdagen als extra informatie.
2. Bij Bestandsinformatie zou een sortering op iets anders dan de naam van de gebruiker een mogelijkheid zijn.
3. Bij de Controle kan het misschien handig zijn een soort van variabele tolerantiegrenzen in te bouwen. De standaard afwijking van 5% zou dan eventueel (per geval) kunnen worden bekeken.
4. Als de zogenaamde 'verkrijgbaarheid' van bepaalde bronnen wordt opgeslagen, levert dit extra mogelijkheden op voor het automatisch plannen. In praktijk zal nu de gebruiker bij een aantal medewerkers standaard een bron invullen zodat die bron gedekt is.

8 Conclusie

Als je vertelt dat je afstudeert bij de werkgever waar je ook je bijbaantje hebt, is de eerste reactie altijd "Oh, dat is mooi". En op zich is het dat ook. Voor mij was het in ieder geval geen optie om ergens anders af te studeren, omdat ik dan niet meer de tijd zou hebben binnen KeyMark mijn taken te vervullen.

Een logisch gevolg van het afstuderen bij je werkgever is dat je de tijd moet verdelen. Dit heeft in mijn geval tot gevolg gehad dat ik weer 45 tot 55 uur per week aan het werk was, om te voorkomen dat het aantal uren die gereserveerd waren voor het afstuderen onder de 35 zou komen.

In de beginfase van het project bleek al snel dat de keuze voor een gespecialiseerd programma een goede was. Er is vrij veel (professionele) software te koop die een goede loonadministratie, een eenvoudige relatiebeheer of een urenplanning of -registratie biedt. Al deze programma's missen echter de flexibiliteit om met klussen te werken, waarbij het aantal uren dus slechts een afgeleide is. Veel van de planningsprogramma's houden zich bovendien voornamelijk bezig met de (business)planning van het bedrijf. Tenslotte blijkt dat veel programma's zijn toegespitst op bouwbedrijven (aannemers), niet echt te vergelijken met KeyMark.

De voordelen zijn geweest dat ik telkens midden in het proces heb gezeten en daarmee veel kennis en ervaring heb opgedaan. Dat heeft me geholpen de applicatie zo effectief mogelijk in het proces te kunnen integreren. Het nadeel daarbij is dat ik zelf ook weet wat er fout kan gaan, en dus bij elke actie ook probeer alle mogelijk voorkomende fouten af te vangen. Een aantal keren moest ik door mijn begeleider even weer met beide benen op de grond gezet worden, zodat ik me weer op het einddoel kon concentreren.

Al met al heeft het langer geduurd dan gepland. Ik heb een maand extra genomen omdat ik een aantal weken niet mijn 35 uur heb gehaald. Een aantal vertragingen in de ontwikkeling (meestal vanwege softwareproblemen) heeft ook een maand extra gekost. Tenslotte heb ik wegens drukte op het werk tussen oktober en december 2000 tijdelijk een pauze ingelast. Deze maanden niet meegerekend komt het neer op een totaal van ongeveer 10-11 effectieve maanden, wat behoorlijk langer is dan de geplande 7.

In deze tijd ben ik regelmatig in situaties gekomen die ik kon oplossen of benaderen dankzij kennis die ik heb opgedaan in mijn studie. In andere gevallen heb ik in mijn oplossing expliciet geprobeerd het probleem te reduceren tot iets wat ik in een eenvoudigere vorm al een keer bij mijn werkgever heb opgelost. Toch heb ik in een paar gevallen de hulp van mijn begeleider en medestudenten moeten inroepen.

Bij het schrijven van het verslag is me duidelijk geworden dat ik de hoeveelheid tijd die daar in gaat zitten niet heb onderschat. Wel heb ik te weinig rekening gehouden met de problemen die bij schrijven kunnen voorkomen. Deze uitten zich voornamelijk in het niet weten wat ik waar wilde zetten, en dan ook nog hoe ik het zo duidelijk mogelijk kon maken. Uiteraard heeft hierbij het feit meegespeeld dat ik niet snel tevreden ben met hoe een en ander beschreven is.

Al met al is het resultaat redelijk bevredigend. Er is een goed doordacht ontwerp gemaakt van het systeem dat een groot deel van de controle op zich kan nemen. De opzet hiervan heeft zich

in praktijktests bewezen en kan langere tijd standhouden, zelfs als de situatie behoorlijk verandert. De implementatie van de applicatie werd bemoeilijkt door praktijkproblemen met onder andere de software. Het maken van een goed bruikbare functionele interface heeft meer tijd gekost dan gepland. Zoals het programma er nu uit ziet, kan er nog maanden worden gesleuteld aan de interface en extra features. De basis is echter gelegd, en kan eenvoudig worden uitgebreid of overgenomen in andere processen binnen de (wederom veranderende) organisatie.