HyperCard en databases

Verharen, E.M.

Publication date:
1989

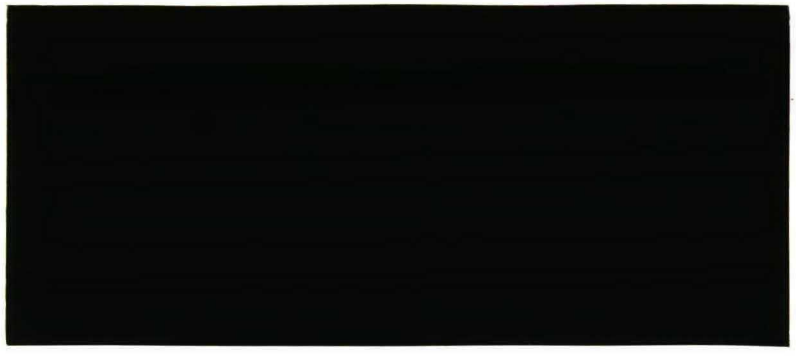
Document Version
Publisher's PDF, also known as Version of record

[redacted]

ITK MEMO

INSTITUTE FOR LANGUAGE TECHNOLOGY AND ARTIFICIAL INTELLIGENCE

INSTITUUT VOOR TAAL- EN KENNISTECHNOLOGIE



I.T.K. Research Memo no. 1
April 1989

HyperCard en databases

Een vooronderzoek naar de mogelijkheden van HyperCard

Egon Verharen

Instituut voor Taal- en Kennistechnologie ITK
Katholieke Universiteit Brabant, Tilburg.

SAMENVATTING.

"*Trying to characterize the kinds of things you can do with HyperCard is like trying to characterize the kinds of things you can do with a deluxe Swiss Army knife* ", Danny Goodman in 'The Complete HyperCard Handbook'.

Dit citaat kenmerkt goed wat HyperCard is of kan. HyperCard is een gereedschapkist waarin elementen zitten waarmee iedereen -van beginner tot ervaren programmeur- op (hun) maat gesneden applicaties kan maken.

Men kan HyperCard zien als een stapel kaarten waarop elke denkbare informatie mag voorkomen, zoals tekst, getallen, tekeningen, dia's of muziek. De kaarten kunnen gegroepeerd worden en ze kunnen onderling op elke willekeurige manier verbonden worden.

HyperCard is ontwikkeld met de bedoeling een implementatie van hypermedia te zijn. Hypermedia is een term die wordt gebruikt voor een informatiesysteem dat alle kennis uit een wereld van interesse bevat en vanaf elke computer geraadpleegd en gebruikt kan worden. De opgeslagen informatie bestaat niet alleen uit tekst, maar ook uit beeld en geluid.

Van alle verschillende applicaties die met HyperCard gemaakt kunnen worden, was de mogelijkheid om HyperCard als database te gebruiken voor mij de meest interessante. HyperCard doorstaat eenvoudig de vergelijking met 'flat file managers', zij het dat ieder z'n specialiteiten heeft. Het maken en afdrukken van rapporten is zeker niet de specialiteit van HyperCard. De vergelijking tussen HyperCard en 'relationele databases' verloopt wat stroever. HyperCard verstaat onder de term 'relatie' een fysieke link tussen gegevens. In deze tekst is verder niet gekeken naar de overeenkomst tussen HyperCard en het netwerk database model.

HyperCard blijkt vanwege zijn gebruikersvriendelijkheid echter uitermate geschikt om als front-end te dienen voor bestaande databases. Voorbeelden zijn gegeven in hoofdstuk 4. Daar HyperCard uit objecten bestaat en werkt met een object-georiënteerde taal (HyperTalk) is het mogelijk om databases ook vanuit een andere dan de flat file of relationele hoek te bekijken. Dit standpunt opent ook meer mogelijkheden om te bekijken hoe een nieuwe, intelligente database te ontwerpen zou zijn in HyperCard.

In de object-georiënteerde programmeertaal HyperTalk zenden de objecten elkaar boodschappen, er is een objecthiërarchie en ook een zekere overerving van handlers. Functies en commando's die niet in HyperTalk gerealiseerd zijn of geschreven kunnen worden, kunnen door de programmeur in C, Pascal of Assembler geschreven worden en als externe commando's of functies aan stacks worden verbonden.

In deze tekst zijn naast het gebruik van HyperCard als database nog vier voorbeelden gegeven van hoe HyperCard te gebruiken zou zijn binnen het ITK. Willen deze ideeën of het onderzoek naar de mogelijkheden van HyperCard voortgezet worden dan is er nog wel nieuwe hard- en software nodig.

Hoewel HyperCard erg flexibel is in het opslaan van alle soorten informatie en erg snel is in het opzoeken van tekst, moeten toch ook de nadelen genoemd worden om een eerlijk overzicht van de mogelijkheden en onmogelijkheden van HyperCard te krijgen. Nadelen zijn onder andere: er kan maar één kaart tegelijk op het scherm, slechts een zeer elementaire script editor, geen debugger, modes en het slechte rapporten maken.

Alles samengenomen blijkt dat HyperCard voor veel doelen (waaronder databases) gebruikt kan worden maar dat de preciese ontwikkeling van een intelligent soort database nog nader onderzocht moet worden.

INHOUDSOPGAVE. blz.

INLEIDING.

De aanzet tot dit schrijven was de vraag van prof. Meersman aan mij of het Apple produkt HyperCard niet te gebruiken was om een intelligent soort database mee te bouwen en wat de verdere mogelijkheden van HyperCard waren. Nu deed ik dit graag, want in de winter van 1987 had dr. Peter Braspenning, toen nog mijn afstudeerbegeleider bij HSA eenzelfde idee, hetgeen door onze drukke werkzaamheden en het gemis van een Macintosh niet verder uitgewerkt kon worden. Hoewel dit idee zich meer toespitste op het gebruik van hypertext (wat hieronder ook besproken zal worden) kwam toch ook het onderwerp van dit rapport aan de orde.

Ik heb geprobeerd in de beschrijving en conclusie zo objectief mogelijk te blijven maar ik moet bekennen dat ik persoonlijk verkikkerd ben op HyperCard. Hoewel de ontwikkeling van HyperCard nog steeds niet helemaal uitgekristalliseerd is, zie ik zoveel nuttige, en ook leuke toepassingen (waarvan een aantal hier ook besproken worden) dat wat voor gevolg dit werk ook zal hebben, ik zelf toch met HyperCard bezig zal blijven.

De tekst is redelijk eenvoudig van opzet. Ik heb dit bewust gedaan omdat prof. Meersman mij tevens gevraagd heeft of ik een informeel praatje wil houden voor geïnteresseerden binnen het ITK over de mogelijkheden van HyperCard en ik deze tekst dan ook als uitgangspunt hiervoor wil gebruiken. In het praatje zullen de voorbeelden van wat met HyperCard mogelijk is nog worden uitgebreid en tevens zal een demonstratie van deze mogelijkheden worden gegeven.

De tekst kan opgesplitst worden in vijf delen. Deel 1 omvat hoofdstuk 1 en 2 die een inleiding geven tot HyperCard. Deel 2 bestaat uit de hoofdstukken 3 t/m 6 die HyperCard en databases behandelen. Deel 3 (hoofdstuk 7 en 8) behandelt de programmeertaal van HyperCard en hoe de functionaliteit van HyperCard vergroot kan worden. In deel 4 (hoofdstuk 9 en 10) worden enkele ideeën voor het gebruik van HyperCard besproken en wat er nodig is om met HyperCard te kunnen werken. Het laatste deel dat bestaat uit de hoofdstukken 11 en 12 tenslotte bevat een opsomming van prestatie en beperkingen van HyperCard en de algehele conclusie over het onderwerp. Zoals gezegd bevat deel 1 een inleiding op HyperCard. Allereerst wordt natuurlijk behandeld wat HyperCard nu eigenlijk is (hoofdstuk 1). Hierin wordt niet een bepaalde definitie gegeven maar meer een verzameling omschrijvingen van wat HyperCard is en kan. Een van die omschrijvingen die naar mijn mening erg belangrijk is, is de beschrijving van HyperCard als implementatie van hypertext of hypermedia. Dit, omdat ik denk dat dit een van de richtingen is waarin de informatie-maatschappij zich in de nabije toekomst beweegt en ook omdat wij binnen het ITK ons toch voor een deel bezig houden met geavanceerde technieken en die toekomst.

hypertext en hypermedia worden in hoofdstuk 2 besproken.

Vervolgens stappen we over naar het eigenlijke onderwerp van deze tekst, namelijk HyperCard en Databases (hoofdstuk 3). Allereerst wordt HyperCard in dit hoofdstuk vergeleken met flat file managers en relationele databases, daarna komen nog algemene punten van databases en in hoeverre HyperCard deze bezit aan bod. In het hoofdstuk over hypertext en hypermedia wordt de parallel getrokken tussen netwerk modellen en hypertext. Ik heb in deze tekst verder niet gekeken naar de overeenkomsten en de verschillen van netwerk databases en HyperCard.

Hoofdstuk 4 geeft een voorbeeld van hoe HyperCard samen met een bestaande database gebruikt kan worden. In hoofdstukken 3 en 4 wordt ook besproken of HyperCard wel de verstandigste keus is om een database in te implementeren en hoe bestaande databases in HyperCard omgezet kunnen worden.

Het hoofdstuk dat uitsluitsel moet geven over de vraag of HyperCard geschikt is om een intelligente database in te implementeren is hoofdstuk 5. Hierin wordt getracht een antwoord te geven op de vraag wat intelligente databases zijn en tevens wordt gekeken of de delen waaruit zij zijn opgebouwd met HyperCard gerealiseerd kunnen worden. In dit hoofdstuk wordt ook aandacht geschonken aan de object-georiënteerdheid van HyperCard.

Hoofdstuk 6 diept dit verder uit en geeft een voorbeeld van hoe een relationele database in HyperCard gerealiseerd wordt en tevens wordt een poging ondernomen om NIAM schema's om te zetten in HyperCard objecten.

In hoofdstuk 7 wordt de kern van HyperCard, namelijk zijn programmeertaal HyperTalk, besproken en tevens wat er allemaal wel en niet mee kan. Hoe dingen op te lossen zijn die niet met HyperCard kunnen, wordt beschreven in hoofdstuk 8 dat aangeeft hoe de functionaliteit van HyperCard vergroot kan worden.

Tussen de omschrijvingen en het deel waar de beperkingen en conclusies behandeld worden heb ik een deel toegevoegd waarin enkele ideeën gegeven worden waarvan ik denk dat ze leuk en nuttig voor het ITK kunnen zijn (hoofdstuk 9). In hoofdstuk 10 wordt gegeven wat er allemaal aan hard- en software nodig is om deze ideeën te verwezelijken en met HyperCard te kunnen werken.

Naast de vele mogelijkheden van HyperCard die door de hele tekst teruggevonden kunnen worden, kent HyperCard (natuurlijk) ook beperkingen. Deze zijn samen met een beschrijving van het prestatievermogen beschreven in hoofdstuk 11.

Tot slot volgt dan de conclusie waarin ik mijn (objectieve) oordeel over HyperCard vel en ook nog enkele conclusies van anderen over HyperCard aanhaal.

N.B. Dit werk is niet vrouw-onvriendelijk bedoeld, waar "hij" of "zijn" staat wordt natuurlijk ook "zij" of "haar" bedoeld. Ook heb ik geen moeite gedaan om overal Nederlandse termen voor

te vinden, dit omdat veruit de meeste boeken over HyperCard in het Engels verschenen zijn en ik daar de terminologie uit geleend heb en omdat de Engelstalige computer-terminologie over het algemeen bij allen die in dit gebied werkzaam zijn ingeburgerd is.

Tot slot wil ik Dick de Reus en Hans Paijmans bedanken voor het correctiewerk en de nuttige opmerkingen.

Hoofdstuk 1.
Wat is HyperCard ?

Het omschrijven van HyperCard is ingewikkeld omdat HyperCard zoveel verschillende gezichten kent. Hieronder zal ik een aantal omschrijvingen van HyperCard geven zoals ik ze in de literatuur gevonden heb.

1.1 Omschrijvingen van HyperCard.

Elke computer draait op en om de verwerking van informatie. Het laat de gebruiker nieuwe informatie creëren, het slaat oude informatie op en het transformeert het ene type informatie in het andere. Zelfs het proces van het beheersen van applicaties en files is een informatie-taak. Laten we deze concentratie van persoonlijke en zakelijke informatie je eigen computer-wereld noemen. Iedere applicatie of documentfile, ieder font, iedere desk accessoire, ze zijn allemaal deel van jouw computer-wereld. Op het laagste niveau geeft HyperCard je de mogelijkheid om je computer-wereld te controleren. HyperCard kan de kern van je computer-wereld worden. Vanuit HyperCard heb je toegang tot elke applicatie en file, het is mogelijk om persoonlijke en/of zakelijke informatie te bereiken die belangrijk genoeg is om op te slaan op disk of file-server. Het is ook mogelijk om toegang te krijgen tot nieuwe informatiebases zoals CD-ROM of veraf gelegen bronnen, verbonden door kabel, telefoon of satelliet.

HyperCard is meer dan alleen een gespecialiseerd programma ontworpen voor speciale taken. Het is ook een gereedschap waarmee men zijn eigen applicaties kan creëren. Het is een krachtig gereedschap dat simpel genoeg is voor nieuwe gebruikers, zij kunnen het gebruiken zodra ze de computer opstarten. Het is echter ook geraffineerd genoeg voor zelfs de meest ervaren programmeur. HyperCard is een hypermedia gereedschapkist (waarover straks meer). Met HyperCard kan iedereen een softwareschrijver worden, maar ook kunnen er applicaties van anderen mee gerund worden. Met HyperCard kan iedereen op zijn eigen manier zijn informatiebehandelingsdroom laten uitkomen. Met HyperCard kan men een database, rekenmachine, grafische dia-show, telecommunicatie-programma of zelfs een nieuw operating system creëren.

HyperCard kan zowel als een omgeving als een medium omschreven worden. De term 'omgeving' wordt gebruikt voor een verzameling regels, condities en mogelijkheden waarmee programma's gecreëerd kunnen worden en waarbinnen ze kunnen werken. Computeromgevingen zijn vaak concentrische schillen om het operating system heen, elk met zijn eigen regels van volgorde en gedrag. HyperCard is bevat in de Macintosh operating system software omgeving welke op zich is bevat in de uitgebreide Macintosh hardware omgeving(en).

Als we met een iets andere bril naar deze feiten kijken

kan HyperCard ook gezien worden als een 'medium', een speciaal apparaat voor communicatie en creatieve doeleinden.

Bill Atkinson, de ontwerper van HyperCard, omschrijft HyperCard als: "*An authoring tool and an information organizer. You can use it to create stacks of information to share with other people, or to read stacks of information made by other people. So it's both an authoring tool and a sort of a cassette player for information*", en ook "*HyperCard is a personal organizer, but that comes as a result of the authoring tool rather than as the motivating factor. You can also use it as a hub for launching documents and applications*" [2].

Het persbericht van Apple dat de bekendmaking van HyperCard begeleidde vermeldt: "*HyperCard is a personal toolkit that gives users the power to use, customize and create new information using text, graphics, video, music, voice and animation. In addition it offers an easy-to-use English-language-based scripting* (programming,EV) *language (called HyperTalk) that gives users an opportunity to write their own programs*" [4].

HyperCard is dus een software creatie die beschreven kan worden als:
- een database of file management programma
- een implementatie van hypermedia
- een programmeertaal, tool set en een hulp bibliotheek
- een koffer vol met handige applicatie programma's
- een tekenprogramma
- een operating system shell
- iets geheel nieuws

Punt 2 zal in het volgende hoofdstuk behandeld worden, en ook over punt 3 is een hoofdstuk opgenomen. In het praatje zullen de punten 4 t/m 7 aan de orde komen en nader worden toegelicht. Eventueel zullen in de volgende paragrafen bovenstaande punten nog terugkomen. De aandacht van dit schrijven gaat echter uit naar punt 1: HyperCard als database.

Zoals hierboven blijkt is HyperCard moeilijk te beschrijven. Het beste is nog om te zeggen dat HyperCard niet elk van deze punten afzonderlijk is maar een produkt dat tegelijk aan al deze omschrijvingen voldoet.

1.2 Hoe ziet HyperCard er uit ?

Tot zover de omschrijving van wat HyperCard is. Voor ons is het ook belangrijk hoe HyperCard er fysiek uitziet.

Men kan HyperCard zien als een stapel (stack) van index kaarten (cards) van 13 bij 19 cm (de grootte van een origineel Macintosh scherm). Een groep gerelateerde kaarten en de file die ze bevat worden samen een stack genoemd. Een kaart kan tekst en grafische informatie in vele en verschillende combinaties bevatten en informatie kan zowel tot de hele stack, een groep kaarten of slecht een enkele kaart behoren. HyperCard slaat tekst op in velden (fields) die van verschillend type (bijvoorbeeld rechthoekig, rond, transparant, ondoorzichtig) kunnen zijn. Een veld bevat tekst van 1 font, grootte en stijl. Een nadeel is dat men font, grootte of stijl van individuele woorden of zinnen niet kan veranderen, alle tekst in een veld moet dus hetzelfde zijn. Een voordeel is dat een van de veldtypes een scrolling text window is zodat een kaart zoveel tekst als men wil kan bevatten (tot 32000 byte). Een ander element van HyperCard is de knop (button). Er zijn ook verschillende typen buttons. Als je op een knop drukt (met de muis op de button klikt) gebeurt er iets, maar daarover meer in de paragraaf over HyperTalk. Een laatste element van stacks is de achtergrond (background). Op deze backgrounds kunnen eigenschappen, velden, buttons etc. verzameld worden die voor alle of een aantal kaarten uit de stack gemeenschappelijk moeten zijn.

Wat HyperCard verder bevat en hoe het wordt beschreven, wordt in het hoofdstuk over HyperTalk gegeven.

Hoofdstuk 2.
Hypertext en hypermedia.

Zoals gezegd is HyperCard een implementatie van hypertext en hypermedia. In dit hoofdstuk zal deze laatste twee begrippen behandelen. De nu volgende tekst is een samenvatting van wat er tot nu toe in de literatuur over hypertext en hypermedia is beschreven. De boeken en artikelen waarop dit verhaal gebaseerd is, staan vermeld in de literatuurlijst en zijn voor het grootste deel te verkrijgen bij mij.

Sinds de tweede helft van 1987 is de term "Hypertext" plotseling zeer populair geworden. In augustus van dat jaar bracht Apple HyperCard uit, in september verscheen het belangrijke overzichtsartikel van Jeff Conklin [8] en in november vond aan de Universiteit van North Carolina de conferentie 'Hypertext 87' plaats, waarvan in [9] zeven lezingen werden gepubliceerd. Ook BYTE [5] wijdde een serie artikelen aan dit onderwerp.

Deze plotselinge opleving ontging ook de wereld van de kunstmatige intelligentie niet (zie Halasz in [9]). Eind 1987 kwam 'KnowledgePro' op de markt, een expertsystem-shell met hypertext faciliteiten. Ook werd tijdens de AAAI-88 een workshop gehouden over AI en hypertext.

De afgelopen jaren kwamen al hypertext-systemen beschikbaar voor workstations, o.a.: gIBIS (Sun), KMS (Sun, Apollo) en NoteCards (Xerox) (zie [5] en [9]). De doorbraak kwam pas in 1987 toen er commerciële produkten voor PC's op de markt kwamen: Guide van OWL en HyperCard van Apple.

Hypertext wordt ingeleid door eerst te vertellen hoe traditionele boeken en computersystemen georganiseerd zijn. Vervolgens worden enkele problemen van deze manier van deze benadering gegeven, waarna hypertext geïntroduceerd wordt. Dit gebeurt in paragraaf 1. In paragraaf 2 wordt een klein stukje geschiedenis behandeld. Ik ga er niet te diep op in omdat het voor dit stuk niet echt interessant is en het uitgebreid gevonden kan worden in de artikelen en boeken uit de literatuurlijst. Vervolgens zullen er enkele definities en omschrijvingen van hypertext en hypermedia gegeven worden. Tevens worden er in paragraaf 3 eisen voor de functionaliteit van een hypertext/hypermedia systeem gegeven en kenmerken waaraan men een hypertext systeem kan herkennen. In paragraaf 4 zullen de meest belangrijke elementen waar een hypertext/hypermedia systeem uit bestaat onder de loep genomen worden. Er is ook een paragraaf over de voor- en nadelen en nog op te lossen problemen van deze nieuwe aanpak opgenomen (paragraaf 5).

Omdat we ons binnen het instituut met kunstmatige intelligentie bezighouden en één van de uitgangsvragen ook was hoe HyperCard gebruikt kon worden voor intelligente taken is paragraaf 6 opgenomen over de mogelijkheden en vergelijking van hypertext en AI. De laatste paragraaf van dit hoofdstuk zal een

inleiding zijn op de rest van dit memo. Hierin wordt bekeken in hoeverre HyperCard aan de kenmerken en eisen voor een hypertext/hypermedia systeem voldoet. We bekijken dus hoe HyperCard een implementatie is van het hypertext/hypermedia concept. Dit zal kort en puntsgewijs gebeuren omdat de er nog genoeg over gezegd wordt in de rest van de tekst.

In het onderstaande zal ik geen onderscheid meer maken tussen hypertext en hypermedia systemen. Vooruitlopend op de definities en omschrijvingen meld ik hier al dat hypermedia een uitbreiding is van het hypertext-concept zodat alle vormen van opgeslagen informatie zoals bijvoorbeeld tekeningen, film, video, geluid, muziek, als ook geschreven tekst eronder vallen. Hierna zal ik dus de termen hypertext en hypermedia door elkaar gebruiken. Dit kan omdat hypertext een onderdeel van hypermedia is en de uitspraken over het algemeen zowel op hypertext als hypermedia van toepassing zijn.

2.1 Waarom hypertext ?

De laatste jaren worden nieuwe dingen niet zo maar ontworpen, meestal is er een goede reden waarom dat nieuwe produkt gemaakt wordt. Er is meestal behoefte aan iets nieuws omdat het oude niet helemaal meer voldoet. Zo ook met hypertext en hypermedia systemen. Hieronder zullen enkele redenen gegeven worden waarom zij nodig zijn.

In het gewone leven komen we een tekst tegen als een papieren document. Tussen onderdelen van zo'n document bestaan een aantal verbindingen. De belangrijkste drie zijn:

a) De onderdelen zijn **altijd** lineair geordend: dit betekent niet anders dan de volgorde waarin de tekstonderdelen achter elkaar staan. Een vorig onderdeel is fysiek verbonden met een volgend onderdeel en omgekeerd;

b) Er is **vaak** een hiërarchische structuur aanwezig: een indeling in hoofdstukken en daarbinnen in paragrafen, etc. Elke paragraaf is b.v. verbonden met het algemene gedeelte van het bijbehorende hoofdstuk en omgekeerd. Maar deze hiërarchische structuur kan slechts gebaseerd zijn op de reeds aanwezige lineaire ordening die de volgorde van hoofdstukken en paragrafen vastlegt. Dit betekent ook dat een document nooit meer dan één hierarchische structuur kent;

c) Er kunnen verwijzingen (references) zijn: het gaat hier over alle verbindingen die niet lineair of hierarchisch zijn. De structuur die verwijzingen aan een tekst geven, is in principe willekeurig en kan daarom alleen maar beschreven worden als netwerk-structuur.

Lineaire ordening is een noodzakelijk gevolg van het feit dat de tekst op papier staat, maar hoeft lang niet altijd voort te komen uit de inhoud van de tekst.

Hiërarchische structuur is een middel om inhoudelijke samenhang aan te geven, en daarmee de raadpleger wegwijs te maken. Maar vaak gaat de complexiteit van de aanwezige samenhang die van een hiërarchie te boven, b.v. omdat er meer hiërarchieën nodig zijn.

Voor de resterende samenhang worden dan verwijzingen opgenomen. De raadpleger die een aantal verwijzingen naslaat, raakt echter al snel het spoor bijster. In de eerste plaats kan het gebeuren dat een gevolgd spoor van verwijzingen te complex wordt om te onthouden, zodat men zijn startpunt kwijt kan raken, en in de tweede plaats doordat het naslaan van verwijzingen zo tijdrovend en omslachtig is. Wanneer echter de tekst niet op papier is vastgelegd, maar op een computermedium, kan de computer op beide punten hulp bieden: zowel het naslaan van een verwijzing als het teruglopen langs een gevolgd spoor van verwijzingen kan dan letterlijk een kwestie worden van een druk op een knop. Men kan naar eigen goeddunken op alle mogelijke manieren door een tekst navigeren; hiervoor wordt vaak de term browsing gebruikt.

In de meeste conventionele papier documenten zijn fysieke en logische structuur dus nauw verbonden. Fysiek is het document een lange lineaire opeenvolging van woorden die voor het gemak verdeeld worden over regels en pagina's. Logisch is het document ook lineair: woorden worden gecombineerd tot zinnen, zinnen vormen paragrafen, paragrafen vormen hoofdstukken etc. Als het document een hiërarchische logische structuur heeft dan wordt deze hiërarchie lineair gepresenteerd: eerst de samenvatting of het overzicht van het geheel dan de inleiding, het eerst hoofdstuk, etc tot aan de conclusie.

Dit geldt ook voor de meeste moderne computer-systemen die zijn opgebouwd uit 'directories' die files bevatten. De files bestaan uit tekst dat is opgebouwd uit karakters. De tekst die in deze hiërarchie is opgeslagen is lineair.

Voor de meeste van onze applicaties is deze lineaire organisatie voldoende. Er komen echter steeds meer applicaties waarvoor een lineaire organisatie niet voldoende is. Bijvoorbeeld de documentatie van een computerprogramma (de niet-executeerbare tekst die de logica van het programma toelicht) is normaal gesproken of tussen de marges van het programma gedrukt, in welk geval het vaak te beknopt is om bruikbaar te zijn, of het is op aparte pagina's tussengevoegd, wat zowel de loop van het programma als de documentatie opbreekt.

In traditionele tekst is niet alleen de organisatie een mogelijk probleem. Ook met de verwijzingen die vaak in een tekst (zeker een wetenschappelijke) voorkomen, zijn er problemen. De problemen met traditionele methodes zoals voetnoten, woordenboeken etc zijn:

- De meeste referenties kunnen niet terug gevolgd worden. Een lezer kan niet eenvoudig vinden waar aan

een specifiek boek of artikel in een document gerefereerd wordt, ook kan een auteur van een artikel niet achterhalen wie aan zijn werk gerefereerd heeft.

- als de lezer zijn weg baant door verschillende verwijzings paden, dan moet hij bijhouden welke documenten hij bezocht heeft en waar hij mee klaar is.
- de schrijver moet annotaties binnen de marges persen of ze in een apart document plaatsen, hetgeen de leesbaarheid niet ten goede komt.
- het volgen van een referentie pad door papieren documenten vergt een fysieke inspanning en vertraagd het werk zelfs als de lezer in een goed georganiseerde bibliotheek bezig is.

Om dit soort problemen of nadelen op te lossen zijn hypertext en hypermedia systemen ontwikkeld.

Er zijn vier soorten hypertext systemen:
1 problem-resolution systemen;
2 on-line browsing systemen;
3 bibliotheek of literatuur-uitwisselings systemen;
4 multi purpose systemen.

ad 1. deze systemen bezitten gereedschappen die je helpen bij het definieren en analyseren van data door gestructureerde types nodes en links. Zij kunnen meestal details onderdrukken door viewing filters.
ad 2. deze systemen bezitten heldere, begrijpelijke scherm-displays voor het presenteren van informatie en makkelijk te hanteren browser commando's om de informatie te onderzoeken.
ad 3. deze systemen bezitten een complexe, meervoudig gestructureerde back-end of database.
ad 4. dit zijn systemen die men naar eigen behoefte aan kan passen aan zijn eigen applicatie of gewoon om mee met hypertext te kunnen experimenteren. HyperCard waar dit memo over gaat behoort tot deze laatste categorie.

2.2 Geschiedenis

De term hypertext is voor het eerst gebruikt in 1965 door Ted Nelson, hoewel de wortels van dit concept al in een artikel van Vannevar Bush uit 1945, waarin een beeld van de toekomst geschetst wordt, gevonden kunnen worden.

"*With hypertext I mean non-sequential writing*", zegt Nelson in zijn inleiding [15]. 'Hyper' staat hier voor uitgebreid en gegeneraliseerd. Later heeft hij dit uitgebreid tot een concept, het Xanadu project geheten, waarin hij computers ziet die al het geschreven materiaal vastleggen in een grote "bibliotheek zonder muren", toegankelijk vanaf elke communicerende computerterminal. Verderop in de tekst geeft hij een nauwkeurigere definitie van hypertext: "*Hypertext is a*

combination of natural language text with the computer's capacity for interactive branching, or dynamic display of a non-linear test which cannot be printed conveniently on a conventional page."

Samen met Nelson is Douglas Engelbart de vader van hypertext. Hij was het die beschreef dat niet alleen de computer een hypertext systeem bepaald. Ook de gebruiker speelt een belangrijke rol. Engelbart's voorgestelde systeem omvatte de menselijke gebruiker als een essentieel element: "*The user and the computer are dynamically changing components in a symbiosis which have the effect of amplifying the native intelligence of the user.*" [16]. Dit is een nog steeds gemeenschappelijke visie onder ontwerpers van hypertext systemen.

Deze twee artikelen hebben de grond gelegd waarop de huidige hypertext hausse is gebouwd en ze zijn daarom aanbevelenswaardig.

2.3 Wat is hypertext en hypermedia ?

De kern van een hypertext systeem kan samengevat worden als:

- zijn mogelijkheid om op hoge snelheid vertakkende operaties op tekstuele 'chunks' uit te voeren;
- een op computers gebaseerd medium voor denken en communicatie;
- hypertext staat toe en bevordert het maken van verwijzingen (zoals voetnoten, commentaar, bibliografische verwijzingen etc, om de schrijver toe te staan om te zeggen: "hier is een gerelateerde gedachte, in het geval dat je geïnteresseerd bent".) en staat het de lezers toe om hun eigen beslissingen te maken over welke link ze willen volgen en in welke volgorde;
- hypertext biedt de mogelijkheid om materiaal in zijn natuurlijke structuur te presenteren in plaats van het in een lineair sequentiële vorm te persen;
- hypertext staat het toe om annotaties op een tekst apart van, maar toch verbonden met, deze tekst op te slaan;
- vanuit het oogpunt van computerwetenschap is de kern van hypertext dat het hybride systeem is dat over traditionele grenzen heengaat:

* hypertext is a database methode die voorziet in een nieuwe manier van direct toegankelijke data.

* hypertext is een representatie schema: het is een soort semantisch netwerk dat informeel tekst materiaal mixt met formelere en gemechaniseerde operaties en processen (zoals het volgen van pointers).

* hypertext is een interface modaliteit: het wordt gekenmerkt door 'control buttons' (link icons) die door de gebruiker willekeurig in de inhoud ingebakken kunnen worden.

Er bestaat (helaas) nog geen algemeen aanvaarde definitie van hypertext: de term wordt gebruikt voor een aantal systemen met grote onderlinge verschillen. Er vallen hier echter wel enige centrale kenmerken te onderscheiden, die hieronder beschreven zullen worden.

We hebben gezien wat Nelson en Engelbart van hypertext zeiden. Nu volgen er nog een aantal omschrijvingen die in de literatuur gevonden kunnen worden.

Gaines en Vickers omschrijven hypermedia in [12.1] als: "*A hypermedia system is one that uses the most advanced technology practically available to facilitate those significant activities that result from increasing the effectiveness of interaction between people and materials relating to knowledge.*"

Janet Fiderio zegt in [5.1] over hypertext: "*Hypertext is op zijn meest elementaire niveau een dbms dat de gebruiker schermen met informatie laat verbinden door gebruik te maken van associatieve links. Op z'n meest geraffineerde, intellectuele niveau is het een software omgeving voor samenwerking, communicatie en kennis acquisitie.*"

En Jeff Conklin omschrijft in zijn belangrijke artikel [8] het concept van hypertext als volgt: Windows op het scherm zijn geassocieerd met objecten in de database en er is voorzien in verbindingen (links) tussen deze objecten, zowel grafisch (als gelabelde tokens) als in de database (als pointers) (zie fig. 1).

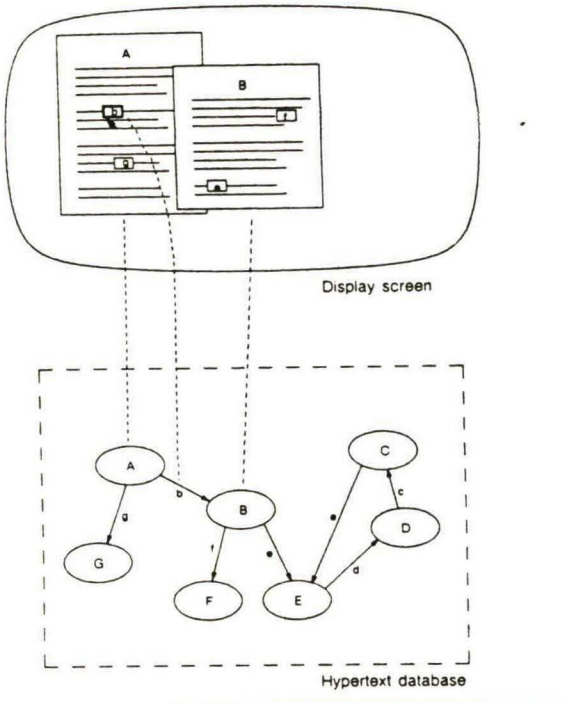


Figure 1. The correspondence between windows and links in the display, and nodes and links in the database. In this example, each node in the hypertext database is displayed in a separate window on the screen when requested. The link named "b" in window A has been activated by a pointing device, causing a new window named "B" to be created on the screen and filled with the text from node B in the database. (Generally, links can have names that are different from the name of the node they point to.)

figuur 1. De correspondentie tussen windows en links op het scherm en nodes en links in de database.

Een essentieel kenmerk van hypertext systemen zijn de machine-ondersteunde links (zowel in als tussen documenten). Het is deze capaciteit van linken die een niet-lineaire organisatie van de tekst toestaat. Een extra kenmerk dat in veel hypertext systemen terugkomt is het veelvuldig gebruik van windows die een 1-1 correspondentie met de nodes in de database hebben.

Een veel gehoorde kreet van sceptici is: Waarom wordt er zoveel ophef over hypertext gemaakt ? We hadden toch al netwerk databases en er zijn toch al manieren om snel informatie uit een database te onttrekken ?

Natuurlijk, maar ik als enthousiasteling vindt dit hetzelfde als zeggen dat de fiets niets nieuws was, omdat het wiel en het licht al bestonden. Echter als men dit combineert in een frame werk en er iets "nieuws" van creëert kan dit vele nieuwe mogelijkheden bieden, zoals de fiets meer vrijheid en bewegingsruimte geeft dan het wiel en het licht an sich.

Terwijl de organisatorische en cross-referentiële structuren van conventionele documenten vast liggen bij de druk, kunnen hypertext links en nodes dynamisch veranderen. Informatie in individuele nodes kan bijgewerkt worden, nieuwe nodes kunnen in de algemene hypertext structuur 'gehangen' worden en nieuwe links kunnen toegevoegd worden om nieuwe relaties te laten zien. In sommige systemen kunnen gebruikers hun eigen links toevoegen om nieuwe organisatorische structuren te maken, om nieuwe documenten van oude te maken. Elk van deze veranderingen representeert een duidelijk verschil tussen hypertext en conventionele documenten, maar wanneer ze samen bekeken worden, produceren ze een kwalitatieve verandering in de manier waarop sommige mensen informatie bronnen conceptualiseren. Het is deze verschuiving in perspectief die zo'n opwinding veroorzaakt en zo'n rijkdom aan nieuwe mogelijkheden in de gedachten van sommigen veroorzaakt.

Ik geef toe dat hypertext meer een ontwikkeld begrip is van wat mogelijk is met een computer dan een totaal nieuw idee. Een soort manuele hypertext is een naslagwerk zoals woordenboeken en encyclopedies. Als men deze ziet als een graaf van tekstuele nodes samengevoegd door verwijzende links, zijn zij oude vormen van hypertext. Intertekstuele referenties zijn niet nieuw. Het belangrijkste van hypertext is eenvoudig dat de referenties machine ondersteund zijn. Machine ondersteuning van links is het binnen het computer netwerk weergeven, lezen, samenwerken, en kritiek geven m.b.v. computer hulpmiddelen.

Zoals ook in hoofdstuk 1 "Wat is HyperCard ?" gedaan is, zal ik nu ook hypertext proberen te beschrijven door te zeggen wat het niet is. Verschillende systemen hebben sommige attributen van een hypertext maar zijn het toch niet:

- window systemen: zij hebben een interface functionaliteit, maar hebben geen enkele onderliggende database.
- file systemen: men zou kunnen beweren dat een file systeem een database is en dat men tussen de nodes (files) beweegt door het aanroepen van een editor met hun namen. Een systeem moet echter een gerafineerder (intellectueler) begrip van links hebben en moet voorzien in meer machine ondersteuning voor de links dan slecht het intypen van de file namen op de een of andere manier.
- de meeste outline processors: er is weinig tot geen ondersteuning voor links tussen outline entiteiten, hoewel hun geïntegreerde hiërarchische database en interface hypertext meer benaderen dan de andere genoemde systemen.
- tekst formaterings systemen: deze systemen staan het toe om tekst fragmenten in verschillende files samen te nemen tot een groot document. Deze structuur is echter alleen hiërarchisch en voorziet niet in een interface voor on-line navigatie binnen het (hoofdzakelijk lineaire) document
- data base management systemen: deze systemen kennen wel verschillende soorten links (bijv. relationele en object-georiënteerde links) maar missen de coherente

interface naar de database die het kenmerk van hypertext is.

Samengevat kan men zeggen dat een hypermedia systeem niets meer is dan een gereedschap voor het creëren, manipuleren, en displayen van nodes van informatie omvat in een netwerk structuur. Voor het systeem zijn alle nodes en links in wezen hetzelfde: het zijn objecten die opgeslagen, onttrokken, gedisplayed, verbonden, etc moeten worden. Voor de gebruiker zijn nodes en links gevuld met betekenisvolle inhoud en georganiseerd in betekenisvolle structuren. Het systeem kan niet direct op deze betekenis opereren. Het voorziet slechts in een collectie tools die gebruikt kunnen worden bij het manipuleren van het netwerk op betekenisvolle manieren. Hypermedia voorziet in een basis voor de integratie van de vele verschillende resources die door de informatie technologie geboden worden in een coherent maar niet restrictief framework. Data processing, simulatie, informatie-onttrekking, tekstverwerking, het presenteren van graphics, en ook databases, kennis banken, expert systemen, en kennis acquisitie kunnen allemaal gezien worden als bijzondere aspecten van hypermedia systemen. De essentiële openheid en algemeenheid van hypermedia moet echter niet leiden tot vaagheid in het concept van systeem ontwerp.

2.3.1 Kenmerken van en eisen voor een hypertext systeem.

Om als hypertext systeem aangemerkt te kunnen worden moet een systeem niet meer dan enkele toetsaanslagen of muis-bewegingen van de gebruiker nodig hebben om een verwijzing te volgen. Een ander belangrijk kenmerk van hypertext is de snelheid waarmee het systeem reageert op verzoeken om verwijzingen te volgen. Een andere eis om een hypertext genoemd te worden is dat de navigatie door een hypertext ruimte computer ondersteund moet zijn.

Het basis idee van hypertext is dat ideeën corresponderen met perceptuele objecten en dat men deze ideeën en hun onderlinge relaties kan manipuleren door het direct manipuleren van windows en icons. Hierover zullen we straks nog meer zeggen in paragraaf 2.4.2.

Kenmerken van (een geïdealiseerd) hypertext systeem zijn dan ook:

* de database is een netwerk van tekstuele (en mogelijk grafische) nodes die kunnen worden voorgesteld als een soort hyperdocumenten.
* windows op het scherm corresponderen met nodes in de database op een 1-1 basis en elke node heeft een naam of een titel die altijd op het scherm getoond wordt. Er zijn echter altijd maar een beperkt aantal windows op het scherm tegelijk open.

* er is in standaard window operaties voorzien: herpositionering, veranderen van grootte, sluiten en het maken tot icon. Positie en grootte van het window of icon zijn aanwijzingen voor de inhoud van het window. Het sluiten van een window zorgt ervoor dat het window van het scherm verdwijnt nadat alle veranderingen die zijn gemaakt in de database-node zijn opgeslagen. Het klikken op een icon zorgt ervoor dat een gesloten window direct wordt geopend.
* windows kunnen een willekeurig aantal link-icons (die in een window een link naar een andere node voorstellen) bevatten. Het link-icon bevat een kort tekst veld dat de inhoud van de node waar het naar wijst suggereerd. Klikken op de link-icon zorgt ervoor dat het systeem de aangewezen node zoekt en onmiddellijk een nieuw window er voor opent op het scherm.
* de gebruiker kan eenvoudig nieuwe nodes creeren en nieuwe links naar nieuwe nodes (voor annotaties, commentaar, uitwerkingen, etc) of naar bestaande nodes (om nieuwe connecties te maken) aanbrengen.
* De database kan onderzocht (gebrowsed) worden op 3 manieren:
    1. door de links te volgen en achter elkaar windows te openen en hun inhoud te onderzoeken;
    2. door in het netwerk (of een deel daarvan) te zoeken naar een string, keyword of attribuutwaarde;
    3. door door het hyperdocument te navigeren gebruikmakend van een browser die het netwerk grafisch presenteert. De gebruiker kan aangeven of de labels van de nodes en/of links moeten worden aangegeven of niet.

Kleine hypermedia databases zijn van minder belang, maar grote interessante databases zijn moeilijk te creëren. Dit werpt een efficiëntie probleem voor hypermedia systeem ontwerpers op dat zij op moeten lossen. Als het te ongemakkelijk is om bij te dragen aan een hypermedia database dan zullen gebruikers dit ook zeker niet doen. Om de ontwikkeling van grote databases te bevorderen moet men: geen mode grenzen kennen tussen editten en navigeren;

- snel nieuwe frames kunnen creëren;
- eenvoudig de structuur kunnen editten;
- voorzien in snelle navigatie. Snelle navigatie is belangrijk, gebruikers moeten snel rond kunnen bewegen om te kunnen komen waar zij willen bouwen. Snelle navigatie is ook belangrijk bij het herstructureren van de database.
- het gebruik van schema's mogelijk maken. Schema's zijn 'chunks' van data die variabele delen bevatten. Schema's kunnen gebruikt worden om data objecten te bouwen die gemeenschappelijke onderdelen hebben,

simpelweg door schema's te copiëren en de variabele delen met de hand in te vullen (templates);
- voorzien in gereedschap voor het importeren van externe databases;
- geen restricties hebben op de grootte van de database;
- database samenvoegingen toestaan.

In [12.1] zijn een aantal eisen ten aanzien van de functionaliteit van hypermedia systemen opgesomd. Essentiële functionaliteit die hypermedia systemen moeten bezitten zijn:
- Integratie. Het medium zelf moet voorzien in verbindingen tussen het materiaal in het medium.
- Vrijheid. De vrijheid van het kunnen browsen door het materiaal langs paden die men niet van te voren heeft aangegeven of die door het systeem worden aangegeven als antwoord op vragen van de gebruiker.
- Flexibiliteit. De flexibiliteit in het kunnen geven van meerdere perspectieven van dezelfde data en het toestaan om nieuwe perspectieven te definieren die niet in de oorspronkelijke database gebouwd waren.
- Bruikbaarheid. De functionaliteit waarin is voorzien moet zo bruikbaar (voor iedereen) mogelijk zijn.

Een hypertext systeem heeft ook de volgende te verwachten functionaliteit:
- diversiviteit. De integratie van diverse auditieve en visuele media.
- uitbreidbaarheid. Het toestaan van het toevoegen van nieuw materiaal en nieuwe perspectieven aan de database.
- gemeenschapsgevoel (sociality). Het toestaan dat meerdere gebruikers toegang hebben tot en veranderingen aan kunnen brengen in de hyperbase.
- ruimtelijkheid (spatiality). Bestaande hypermedia systemen maken sterk gebruik van de ruimte metafoor. Informatie wordt opgemaakt in de ruimte en verbindingen worden gezien als paden door de ruimte die via gateways gevolgd kunnen worden. Een groot deel van de mensheid gebruikt deze metafoor en visueel denken voor het oplossen van problemen.

Gewenste functionaliteit van hypermedia systemen is:
- programmeerbaarheid. Een hypermedia systeem moet voorzien in een buitengewoon krachtige user interface, dat voorziet in intelligente help- en presentatie faciliteiten tot elke gewenste diepte.
- oriënteerbaarheid. Overzicht faciliteiten moeten aanwezig zijn om het verdwaald-in-de-hyperruimte-probleem te verminderen of te elimineren.
- bestuurbaarheid. Een systeem zonder modes, dat iedere mogelijke keuze voor de gebruiker beschikbaar maakt,

kan moeilijk te gebruiken zijn omdat het een aanzienlijke last (die van het beslissingen maken) op de schouders van de gebruiker legt. Het hypermedia systeem moet voorzien in verschillende vormen van leiding geven, gidsen die het kiezen niet beperken maar juist een goede basis voor de keuze leggen, de meest aannemelijke aangeven en de redenen hiervoor.

- hercreëerbaarheid. Om zijn perspectieven te optimaliseren kan de gebruiker gebruikmaken van de vrijheid om door het systeem te browsen op zoek naar de gewenste informatie. Het is echter mogelijk dat het uiteindelijke perspectief minder gewenst is dan een eerdere en het is daarom belangrijk dat hij terug kan keren naar zo'n perspectief.
- communiceerbaarheid. Het is nooit duidelijk voor een systeemgebruiker of het materiaal dat aan een andere gebruiker gepresenteerd wordt wel begrepen is. Soms is het erg belangrijk en wenselijk dat er voorzien is in gateways naar materiaal dat het begrip van ander materiaal test.

2.4 Elementen van hypertext systemen.

Na hypertext omschreven te hebben kijken we nu naar de delen waaruit een hypertext systeem is opgebouwd. Allereerst zullen we de links behandelen, vervolgens de nodes en tot slot nog elementen als de browser.

2.4.1. Links

Links kunnen voor verschillende doeleinden gebruikt worden:

- verbinden van een documentreferentie met het document zelf;
- verbinden van een annotatie of commentaar aan de tekst waar het voor geschreven is;
- het voorzien in organisatorische informatie (bv. om de relatie tussen twee stukjes tekst kenbaar te maken of tussen een inhoudstabel en de secties);
- verbinden twee opeenvolgende stukken tekst of een stuk tekst met al zijn onmiddellijke opvolgers/nakomelingen (hiërarchische structuur aanbrengen);
- verbinden van punten in een tabel of figuur met langere beschrijvingen of andere tabellen of figuren.

Er zijn twee manieren om twee punten expliciet te verbinden:

1) door referentie. Dit is niet hiërarchisch, gericht (ook terug), en het link domein en bereik kan een enkel punt of stuk tekst zijn of
2) door organisatie. Dit implementeert hiërarchische

informatie, het vormt een tree subgraaf in de hypertext netwerk graaf (dit correspondeert met de superconcept link van semantische netwerken.

De meeste hypermedia systemen voorzien in verschillende linktypes. In sommige systemen zijn er alleen voorgedefinieerde links, in andere systemen kunnen ze door de gebruiker gedefinieerd worden, soms zijn links ook objecten met interne structuren. De type informatie kan visueel aangeduid worden door labels, maar kan ook verborgen zijn. Het belangrijkste doel van link types is om gebruikers en programma's te voorzien van meer informatie over de bestemming van de link.

Links moeten twee kwaliteiten hebben:

- de computer moet ze kunnen volgen en opsporen
- ze moeten je snel van de ene naar de andere node kunnen brengen.

Een getypeerde link specificeert een bepaalde relatie tussen twee nodes, een die men zelf kan specificeren. Links kunnen echter meer dan alleen twee nodes verbinden. Afhankelijk van het hypertext systeem kunnen links annotaties verbinden aan een document en voorzien in organisatorische informatie. Daarom kunnen links gebruikt worden voor het definieren van de relaties van een node met andere nodes binnen de database.

De meeste systemen die tot nu toe ontworpen zijn, hebben de hiervoor beschreven typen links. In een volgende generatie systemen zouden deze uitgebreid kunnen worden met bijvoorbeeld: clusterlinks (een link die meer dan twee nodes verbindt), attribuut/waarde paren (men kan dan ook op waarde van een attribuut zoeken), procedurele aanhangsels (die een bij effect produceren bij het linken).

2.4.2. Nodes

Veel van de kracht van hypertext zit in de mogelijkheid van het linken: het zijn de machine verwerkbare links die een tekst uitbreiden buiten de enkele lineaire dimensie. Wat betreft de nodes van hypertext: als hypertext wordt gebruikt voor denken, schrijven en ontwerp gereedschap kan er een natuurlijke correspondentie ontstaan tussen objecten in de wereld en nodes in de hypertext database. Door gebruik te maken van dit object-georiënteerde aspect kan een hypertext gebruiker een flexibel netwerk bouwen dat zijn probleem (of oplossing) modelleert. Vanuit dit gezichtspunt zijn de links minder belangrijk dan de nodes. Zij vormen slechts de lijm die de nodes bij elkaar houdt, maar de nadruk ligt op de inhoud van de nodes. Het opdelen van ideeën in units die nodes in een hypertext kunnen zijn gebeurt geheel onder de verantwoordelijkheid van de schrijver en dit proces is een kunst. Het proces van het identificeren van een semantische eenheid (zoals een idee of concept) met een syntactische (zoals een paragraaf of hypertext node) is niet uniek voor hypertext. Sommige hypertext systemen

verdelen nodes in verschillende types. Deze getypeerde nodes kunnen extreem handig zijn, vooral als men overweegt hen een interne structuur te geven, omdat de types gebruikt kunnen worden om de verschillende gestructureerde vormen uit elkaar te houden.

Een ongetypeerde node is een doos voor informatie. Het heeft geen label of descriptor, dus men kan het vullen met allerlei informatie in een willekeurige vorm. Een getypeerde node is voorzien van een label en de descriptor helpt om de stijl van informatie die de node kan bevatten te bepalen. Types helpen de gebruiker om de nodes te classificeren of om gespecialiseerde commando's de definieren.

Een link kan dan ook gezien worden als een component van een node.

2.4.3. Structuur en browsers.

Een van de onderdelen van het geïdealiseerde hypertext systeem uit paragraaf 2.4.1 was de browser. De browser is een belangrijke component van een hypertext systeem. Als het hyperdocument groter groeit en complexer wordt, wordt het voor een gebruiker steeds makkelijker om gedisoriënteerd of verdwaald te raken, zie voor dit orientatie-probleem ook de volgende paragraaf.

Verdwaald zijn betekent: niet weten waar je bent in de informatie ruimte, niet weten hoe iets te bereiken waarvan je denkt dat het bestaat, ergens in een document aankomen en vergeten zijn wat je daar moest doen, vergeten om terug te keren van zijpaden of niet verder gaan met eerder geplande zijpaden, niet weten of er nog andere relevante frames in het document zijn, vergeten welke secties bezocht en/of veranderd zijn, de onmacht om een coherent beeld van de onderzochte en gewijzigde frames te vormen na uren browsen. De meeste gebruikers van grote hyperdocumenten staan erop een of ander mechanisme te hebben om de inhoud van nodes, links of regions te scannen, ofwel naar geselecteerde keywoorden of naar willekeurige strings.

Sommige systemen voorzien in speciale voorzieningen voor hiërarchische structuren. Aan de ene kant is abstractie een fundamenteel cognitief proces, en hiërarchische structuren zijn de meest natuurlijke structuren voor het organiseren van niveaus van abstractie. Aan de andere kant zijn er duidelijke gevallen waarbij niet-hiërarchische links zijn gewenst. Het voordeel van hiërarchische structuren is dat de commando-taal voor het navigeren heel simpel is. Het nadeel is dat de structuur een functie van de paar specifieke criteria is die gebruikt werden om de structuur te bouwen. De oplossing hiervoor is om het toe te staan dat de informatie-elementen in meerdere hiërarchiën gestructureerd kunnen worden.

2.5 Voordelen en problemen van hypertext systemen.

In deze paragraaf zullen puntsgewijs voordelen van het gebruik van hypertext systemen behandeld worden. Natuurlijk kan men zich geen beeld vormen van hypertext systemen als niet ook de nadelen en nog op te lossen problemen vermeld worden. Tot slot van deze paragraaf wil ik een korte conclusie over dit onderwerp geven.

2.5.1. Voordelen.

Als voordelen van het gebruik van hypertext kunnen genoemd worden:

- nieuwe mogelijkheden voor authoring (schrijven) en ontwerpen. Authoring is het ontwerpen van een document en men heeft hierbij te maken met het structureren van ideeën, de volgorde van presentatie en de conceptuele verkenning van het onderwerp.
- nieuwe mogelijkheden voor lezen en onttrekken van gegevens. Het volgen van referenties is eenvoudiger en met de nieuwe zoektechnieken kunnen snel de juiste gegevens uit de database worden onttrokken.

Als operationele voordelen kunnen genoemd worden:

-- de eenvoud van het volgen van referenties. Machine ondersteuning voor het volgen van links betekent dat alle verwijzingen allemaal even eenvoudig te volgen zijn naar hun referent als terug naar hun referentie.
-- de eenvoud van het creëren van nieuwe referenties, het is bijvoorbeeld eenvoudig het document van iemand anders van annotaties of commentaar te voorzien zonder zijn document echt te wijzigen.
-- informatie structurering. Zowel hiërarchische als niet-hiërarchische organisatie kan opgelegd worden aan ongestructureerde informatie.
-- globale gezichtspunten (views). Browsers voorzien in een inhoudsopgave stijl, die eenvoudiger herstructurering van grote of complexe documenten ondersteunen. Globale en locale (node of pagina) views kunnen effectief gemixed worden
-- op maat gemaakte documenten. Tekstfragmenten kunnen samengenomen worden op verschillende manieren zodat hetzelfde document meerdere functies kan hebben.
-- modulariteit van informatie. Omdat hetzelfde tekstsegment vanuit verschillende plaatsen gerefereerd kan worden kunnen ideeën met minder overlap en duplicatie uitgedrukt worden. consistentie van informatie. Verwijzingen zijn ingebakken in een tekst en als de tekst verplaatst wordt (misschien zelfs naar een ander document) dan geeft de link informatie nog steeds direct toegang tot de referentie.

-- taak stapeling. In sommige systemen wordt de gebruiker ondersteund in het tegelijk onderhouden van meerdere paden van onderzoek die ook tegelijk op het scherm te zien zijn, zo dat ieder pad teruggevolgd kan worden tot de oorspronkelijke taak.
-- samenwerking: meerdere auteurs kunnen samenwerken waarbij het document en de commentaren over het document sterk verbonden kunnen zijn.

2.5.2. Nadelen en problemen.

Nu de huidige generatie hypermedia systemen steeds algemener gebruikt gaat worden worden de beperkingen van deze systemen steeds duidelijker. Het simpele node en link model is niet rijk en compleet genoeg om informatie representatie management en de presentatie taak nodig voor vele applicaties te ondersteunen .

Er zijn twee klassen problemen :

1) die met de huidige implementaties: zoals de vertraging in het weergeven van gerefereerd materiaal, beperkingen op namen en andere eigenschappen van links, het gebrek of gemis van browsers of onvolkomenheden in de browsers, en
2) problemen eigen aan hypertext:

a. disoriëntatie, dit is het eerder genoemde "lost-in-hyperspace-probleem", de neiging om begrip van locatie en richting in een niet-lineair document te verliezen;
b. cognitieve overhead: de toegevoegde inspanning en concentratie die nodig zijn om verschillende taken en paden tegelijkertijd te onderhouden.

ad a. samen met de mogelijkheid om informatie complexer te organiseren ontstaat het probleem van weten waar je bent in het netwerk en hoe je naar een andere plaats waar je het bestaan van weet (of denkt te weten) moet komen. In lineaire tekst heeft een lezer twee mogelijkheden: het was eerder in de tekst of verderop in de tekst. In hypertext zijn er echter meer dimensies waarin men zich kan begeven en dus is het eenvoudiger om gedisoriënteerd of verdwaald te raken. Oplossingen voor dit probleem kunnen geboden worden door:

- grafische browsers (ook voor delen van het netwerk). Een adequate doorzichtigheid is moeilijk te onderhouden voor een groot en complex hypertext netwerk vanwege:
  * het grote aantal nodes en links;
  * de frequente veranderingen aan het netwerk;
  * langzame of moeizame reactie op de control inputs van de gebruiker;
  * onvoldoende visueel onderscheid tussen nodes en links;
  * niet visueel georiënteerde gebruikers combineren er maar op los en maken het

practische onmogelijk om de disoriëntatie alleen met browsers op te lossen;
- query/search mechanismen;
- filteren van informatie tot een handelbaar niveau van complexiteit en detail.

ad b. het is moeilijk om te wennen aan de extra mentale overhead die nodig is voor het creëren, naam geven en bijhouden van links.

Hypertext is een zich ontwikkelende technologie die nog veel op te lossen problemen kent:

- misschien wel het moeilijkste deel van het creëren van een hypertext systeem is niet het bouwen van een interface maar het creëren van een degelijk onderliggend datamodel dat onderhouden kan worden.
- een ander probleem voor sommige gebruikers is dat sommige hypertext systemen je controle geven terwijl leiding (guidance) meer op zijn plaats is. Terwijl grafische browsers kunnen helpen bij het niet verdwaalt raken, kan het gebrek aan visuele en spatiële aanwijzingen disoriënterend werken.
- een ander ingewikkeld punt is het opbreken van een gedachte of een informatie segment in nodes. Thema's in een document of gedachte kunnen sterk samenhangen, zo zelf dat het opbreken van de informatie in discrete nodes verstorend zou werken (daarom is ook niet alle literatuur geschikt om om te zetten naar hypertext). En zelfs als een document of gedachte discrete componenten heeft, kan men zich niet op een niveau bevinden waar men deze units kan overzien als men een hypertext applicatie maakt. In zulke gevallen kan men informatie te vroeg opbreken in nodes en er later achter komen dat dat verkeerd was. Dan moet men de informatie editten, herarrangeren of hernoemen. Virtuele structuren zouden handig zijn in deze situatie. Zij zouden dynamisch veranderd worden wanneer men nodes en links toevoegt of verwijdert. Virtuele structuren zijn gelijkwaardig aan relationele database management views.

2.5.3. Conclusie.

Hypermedia kunnen gezien worden als een belangrijke fase in de evolutie van media in het algemeen en in dat van de informatie-technologie als een medium voor menselijke kennis in het bijzonder. De nieuwe mogelijkheden voor interactieve communicatie met passieve en actieve opslagplaatsen voor kennis die geboden worden door computersystemen van een nieuwe generatie maken het mogelijk om het menselijk intelligent handelen verder te verfijnen. Conklin zegt hierover: *"People who think for their living must contend with the fact that the brain can create ideas faster than the hand can write them or the mouth can speak them. There is always a balance between refining*

the current idea, returning to a previous idea to refine it, and attending to any of the vague 'proto-ideas' which are hovering at the edge of consciousness. Hypertext simply offers a sufficiently sophisticated 'pencil' to begin to engage the richness, variety and interrelatedness of creative thought." [8].

2.6 Hypertext en AI.

Een interessante richting van onderzoek is de integratie van hypermedia en AI. In veel opzichten passen hypermedia en op kennis gebaseerde systemen bij elkaar. In het bijzonder zijn hypermedia systemen, frame-based systemen en object-georiënteerde systemen op een hoog niveau van abstractie bijna identieke data modellen. Elke van deze technologieën is gebaseerd op een begrip van getypeerde entiteiten die een netwerk structuur via inter-entiteit referenties vormen. De technologieën die gekozen zijn voor verdere ontwikkeling verschillen in de specifieke aspecten van dit basis model. Hypermedia richt zijn aandacht op zwaar gewichtige entiteiten (hele documenten) en inter-entiteit relaties (links). Object-georiënteerde systemen richten zich op het definiëren van een type (class) hiërarchie voor de entiteiten en de operaties (methods) die op instanties van elk type uitgevoerd kunnen worden. Op frames gebaseerde systemen richten hun aandacht op punten zoals overerving en defaults voor slot waarden, alsook op de integratie van truth maintenance, inference engines en op regels gebaseerd redeneren met de frame representatie.

Het is duidelijk dat het lenen van ideeën van de op object- en frames gebaseerde technologieën vruchtbaar zou kunnen zijn voor de vooruitgang van hypermedia systemen. Het samengaan van het object-georiënteerde paradigma en hypermedia is voorgesteld in:

*) Hara en Kaneko- A new multi-media electronic book and its functional capabilities;
*) Hirano Hypermedia-based documentation system for the office environment,

(allebei in RAIO-88, zie [13]);

*) Woelk, Kim en Luther An object-oriented approach to multi-media databases, proc acm sigmod 86;
*) Fishman et al IRIS: an object oriented dbms, trans ois vol 5 nr 1 jan 87;
*) Caplinger an information system based on distibuted objects, proc oopsla 87.

Het hypertext-idee om een gerichte graaf van informele tekst elementen te maken is gelijk aan het AI concept van semantische netwerken. Een semantisch netwerk is een kennisrepresentatie-schema bestaande uit een gerichte graaf waarin concepten gerepresenteerd worden door nodes en de relaties tussen concepten door links. Wat een semantische netwerk als een AI-representatie schema onderscheidt, is dat concepten in de representatie geindexeerd worden door hun semantische inhoud ipv door een willekeurige (bv alfabetische)

volgorde. Een voordeel van semantische netwerken is dat zij natuurlijk zijn om te gebruiken, omdat gerelateerde concepten de neiging hebben om samen te kruipen in het netwerk. Ook is een oncompleet of onconsistent gedefinieerd concept makkelijk te ontdekken omdat in een betekenisvolle context is voorzien door de omliggende concepten waar het aan vast is gelinked. De analogie met hypertext is rechttoe rechtaan. Men kan aan hypertext nodes denken als zijnde alleenstaande concepten of ideeën representerend, terwijl internode links semantische afhankelijkheden tussen deze ideeën representeren en het proces van een hypertext netwerk bouwen als een soort van informele kennis engineering. Het verschil is dat AI kennis engineers streven naar het bouwen van representaties die mechanisch geïnterpreteerd kunnen worden, terwijl het doel van een hypertext schrijver vaak het vangen van een verweven ideeën is zonder naar de machine interpreteerbaarheid te kijken. Het werk met semantische netwerken verondersteld ook enkele natuurlijke uitbreidingen van hypertext zoals: getypeerde nodes, semi-gestructureerde nodes en overervings hiërarchiën van nodes en link types.

De relatie tussen systemen die kennis structureren (kennissystemen) en hypertext wordt op de volgende manier door C.A. Benschop in [14] omschreven: "*Hypertext is een manier om structuur te geven aan kennis (om de structuur van kennis weer te geven, EV). Structuur is meta-kennis, kennis over kennis, en deze meta-kennis kan kennis toegankelijk, en dus bruikbaar maken. Hypertext is daarom een manier om kennis te presenteren. Kennissystemen hebben dezelfde pretentie. De vraag dient zich dus aan, wat de relatie is tussen hypertext en kennissystemen.*

Hypertext, in de tot nu toe gangbare vormen, wordt gebruikt voor:

* *het ondersteunen van kennisstructurering;*
* *het presenteren van gestructureerde kennis ter raadpleging.*

Kennissystemen (we beperken ons hier tot kennissystemen voor menselijke raadplegers):

* *bevatten gestructureerde kennis;*
* *redeneren aan de hand van die gestructureerde kennis, in meer of mindere mate in samenwerking met de raadpleger.*

In kennissystemen gaat het om structureren van kennis, in hypertext om het raadplegen van kennis. Bij het raadplegen kan het redeneren zowel door de raadpleger als door het systeem plaatsvinden. Een geïntegreerd kennis-hypertext-systeem stelt de raadpleger in staat om overzichtelijk met de opgeslagen informatie om te gaan."

2.7 HyperCard als implementatie van hypertext/ hypermedia.

In de vorige paragrafen hebben we hypertext als concept beschreven en ook een aantal kenmerken en eisen die hypertext/hypermedia systemen zouden moeten hebben gegeven. In deze paragraaf zullen we zien in hoeverre HyperCard hieraan voldoet. Deze paragraaf zal niet uitputtend zijn, omdat de rest van dit memo over HyperCard gaat en vele punten die hier niet genoemd zullen worden, zullen daarin duidelijk gemaakt worden.

Allereerst zullen we kijken naar de kenmerken (zie ook paragraaf 3).

HyperCard kan gezien worden als een soort database van nodes die kunnen worden voorgesteld als hyperdocumenten. HyperCard windows corresponderen met deze nodes op een 1-1 basis. In HyperCard kan er slechts één window tegelijk open zijn. Niet alle standaard window operaties kunnen hierop uitgevoerd worden. Er kan bijvoorbeeld geen icon van het window gemaakt worden. Wel kan elk window (elke node) een willekeurig aantal link icons bevatten. Deze link-icons werken precies zoals voorgesteld. Het werken met HyperCard en dus ook het creëren van nieuwe nodes en links is, in overeenstemming met de Apple filosofie, heel eenvoudig gehouden. HyperCard mist wel de grafische browser. Deze kan wel zelf gemaakt worden maar hij wordt niet automatisch gegenereerd. Er is wel voorzien in speciale navigatie commando's en text en string search. Bij het maken van objecten (nodes, links) en het navigeren moet men wel een mode grens tussen het editten en het navigeren oversteken. HyperCard links hebben wel een interne structuur en kunnen ook van verschillend type zijn.

De linksource (het domein) van een link is in HyperCard een frame (card) of een object in een frame (button, field) of een individueel object (woord op card). De linkdestination (het bereik) is een heel frame (card). Omdat HyperCard erg snel een link kan volgen, denken we aan HyperCard als iets dat de tijd dimensie gebruikt om verbonden nodes bij elkaar te houden (time multiplexing) in plaats van te proberen ze allemaal tegelijk op het scherm te laten zien (space multiplexing).

HyperCard voorziet maar 1 node type. Er is in variëteit voorzien op het niveau van individuele items binnen een node (velden, text, graphics, beelden, buttons).

HyperCard is voor Personal Computers, dus geen multi-user access, wel is het mogelijk dat verschillende mensen een voor een aan hetzelfde document werken.

Volgens Halasz is HyperCard niet een echt hypermedia systeem (al hoewel het sommige hypermedia kenmerken heeft). Maar het heeft wel het soort uitbreidbaarheid (extensibility) en werkbaarheid (tailorability) dat huidige systemen niet hebben maar standaard moet zijn in toekomstige generatie hypermedia systemen.

Meer over HyperCard zelf en zijn voor- en nadelen kunt u vinden in de volgende hoofdstukken.

Voor verdere informatie en een uitgebreide bespreking van hypertext verwijs ik naar de BYTE van oktober 1988 [5] waarin vier artikelen over hypertext plus een lijst met overige hypertext boeken en artikelen, commerciële produkten en instituten waar onderzoek naar de mogelijkheden en moeilijkheden van hypertext plaatsvindt, zijn opgenomen en naar de artikelen (o.a. Jeff Conklin) en hun literatuurlijsten in de IEEE Computer van september 1987 [8].

Hoofdstuk 3.
HyperCard en database systemen.

Ondanks het feit dat HyperCard een eigen tint aan het conventionele database-concept geeft, kan het ook (zij het incompleet) omschreven worden als een databaseprogramma.

Hieronder zal ik eerst samenvatten wat er in de boeken over gezegd wordt en dan mijn eigen gedachten daar over geven.

Dit hoofdstuk begint met een paragraaf waarin en stukje evolutie van databases wordt gegeven en waarin de huidige databases grofweg worden opgesplitst in twee categorieën. Vervolgens wordt in paragraaf 2 besproken hoe HyperCard in deze categorieën past. In paragraaf 3 komen overige eigenschappen van databases en database-ontwikkeling aan de orde en ook weer hoe HyperCard deze behandeld. In de laatste paragraaf van dit hoofdstuk staat de vraag of en wanneer men nu HyperCard of de traditionele databases moet gebruiken centraal.

3.1 De evolutie van databases.

In [1] gaat men uit van een "losse", algemene definitie die een database programma beschrijft als een programma dat gestructureerde verzamelingen informatie opslaat, organiseert en manipuleert (*"In short, it keeps lists"*[1]). Natuurlijk moet een databaseprogramma meer kunnen dan alleen een lijst (of lijsten) opslaan (men kan ook met een word-processor lijsten intypen en opslaan). Op z'n minst moet een databaseprogramma toestaan om de lijst(en) te sorteren en om elementen uit een lijst te selecteren op een of andere methodologische manier. Verder wordt in [1] gegeven dat sorteren en selecteren de belangrijkste taken van een database programma zijn, de rest is slechts uitbreiding en specialisatie hiervan. Een bekende uitbreiding is de mogelijkheid om de "view" op de data (de opgeslagen lijsten) te specificeren en veranderen. Een andere uitbreiding is de mogelijkheid om feiten anders dan expliciet opgeslagen in de lijsten te verzamelen en deze in geformatteerde rapporten af te leveren.

Hoe gestructureerder een lijst van informatie is, hoe simpeler en kleiner een programma dat de lijst moet bewerken kan zijn. De eerste databases eisten dat de gebruiker vooraf precies specificeerde welke items opgeslagen moesten worden, hoe groot ieder item was en wat voor soort informatie het zou bevatten (bijvoorbeeld een tekst, getallen, datum, etc.).

Latere databases zwakten dit af en eisten steeds minder een vooraf gedefinieerde structuur van de informatie die zij bewerkten. Enkele van deze nieuwe programma's maken het zelfs mogelijk om informatie op te slaan waarvan het aantal items per ingang in de lijst varieert per ingang.

Een volgende stap in de evolutie was het relationele concept. Ook hiervan wordt in [1] een simpele definitie gegeven: *"Relationeel structureren is een manier om te voorkomen dat bepaalde informatie meerdere keren in de lijst wordt opgeslagen"*. Hieronder zal dit (slechts redundantie-beperkende aspect) toegelicht worden.

Drie andere trends in database programma's zijn:
- de mogelijkheid om data te beheersen die meer naar free-form tekst neigt dan naar gestructureerde lijsten van items van vaste lengte;
- de mogelijkheid om zowel grafische informatie als tekst op te slaan;
- database programma's waarvan de datamanipulatietaal volledige programmeertalen zijn.

Bijna geen enkel feit bestaat in een vacuum. Het probleem is dat als men een informatiebron voor een gewenst feit vindt, die bron vaak niet de extra informatie bevat die men nodig hebt. Informatie bestaat vaak uit een feit en vele draden die daaraan verbonden zijn en naar vele richtingen gaan. Aan het eind van iedere draad is een nieuw feit, met ook weer vele draden die weer in andere richtingen gaan. Als zo'n web van informatie groeit wordt het steeds moeilijker om een bepaald item te lokaliseren. Het nadeel hiervan is dat elk informatie-niveau dat we moeten doorlopen op zoek naar een bepaald feit, het verlangen om het zoeken daadwerkelijk uit te voeren verkleint.

Door de manier waarop we opgegroeid zijn met computers en hun programma's zijn we (te) vaak geneigd om ieder document als een apart stuk informatie te zien. En, tenzij we in een volledig geïntegreerde applicatie-omgeving werken, kijken we naar een bepaald document alsof het geen betrekking heeft op iets buiten het huidige applicatieprogramma. Het is echter belangrijk te onderkennen dat er maar weinig opgeslagen documenten zijn die in hun privé vacuum bestaan. Vaker is een document slechts een deel van informatie met een veel grotere context. Het is dan belangrijk om deze verbintenis expliciet te kunnen maken.

Een bekend type informatie management software is de categorie DataBase Management Software (DBMS). Binnen de categorie van personal computers zijn er twee typen database management:
- file management;
- relationeel database management.

3.1.1 File management.

'File' staat hier voor een eindige collectie informatie, normaal een duplicaat van een lade vol ingevulde formulieren. Voor men de database (file) kan gebruiken, moet men een on-screen formulier creëren dat iedere keer ingevuld moet worden

als nieuwe informatie aan de database wordt toegevoegd of dat automatisch ingevuld wordt als informatie uit de database opgevraagd wordt.

Een taak van de file management software is het bieden van manieren om de formulieren te sorteren in elke volgorde die de gebruiker wenst. Een andere belangrijke functie van file management software is het zoeken. Goede file management software maakt het mogelijk om meerdere of meervoudige zoekcriteria op te geven. Daarna moet het resultaat van het zoeken op een door de gebruiker gespecificeerde manier afgedrukt kunnen worden.

3.1.2 Relationeel database management.

Een ingewikkelder type database management is het relationeel database management. [2] beschrijft relationeel als de mogelijkheid om links (relaties) tussen individuele files te leggen. De relationele database management software kent het zelfde soort sorteren, zoeken en rapporteren als file managers, met dit verschil dat de mogelijkheid wordt geboden om informatie uit meerdere databases (files) te halen. Hiermee wordt het invullen van de formulieren van de file managers sterk vereenvoudigd.

3.2 Hoe past HyperCard in de database categorie ?

HyperCard voldoet volledig aan de voorgaande, zij het ruime, definities en beschrijvingen van een database en wat het moet kunnen. (Deze definities komen uit boeken over HyperCard en men moet er dus op bedacht zijn dat ze weleens zo gegeven kunnen zijn dat HyperCard er aan voldoet).

Voorbeelden hiervan zijn:

- HyperCard heeft goede basismogelijkheden voor het opslaan en organiseren van lijsten van gestructureerde informatie;
- HyperCard is ook redelijk bedreven in het sorteren;
- HyperCard bevat ingebouwde functies, zij het erg beperkte (zie ook de paragraaf over Beperkingen), voor het maken van rapporten;
- HyperCard is een ster in het verbindingen leggen tussen items in gerelateerde lijsten;
- Het biedt simpele handelingen in de initiële ontwerpstappen van het organiseren van een database en ook het reorganiseren van het ontwerp is niet ingewikkeld;
- HyperCard gaat verder in de trend waar minder gestructureerde data toegestaan is door opslag van informatie toe te staan met weinig tot geen voorafgaande zorg voor grootte, type, of volgorde;
- HyperCard is in staat om blokken van veranderbare grootte van free-form tekst op te slaan en snel dingen in deze tekst te vinden;

- Bovendien kan HyperCard informatie in de vorm van plaatjes, geluid, animatie, etc. opslaan en is verbinding met video, etc. mogelijk;
- HyperCard is geschreven in HyperTalk en als we HyperCard als database zien dan past deze taal dus in de trend van database programma's die ook programmeertaal zijn.

De inleiding in database management software in de vorige paragraaf was nodig om aan te kunnen geven hoe in [2] wordt bekeken hoe HyperCard in deze categorieën past.

Allereerst wordt in [2] opgemerkt dat HyperCard niet is bedoeld om huidige database management software te vervangen. Het zal zelfs zo zijn dat HyperCard samen met een database programma gebruikt wordt (zie hierover ook de volgende paragraaf). Het volgende vergelijkt HyperCard met bovenstaande omschreven database management software. Hierin zullen ook punten besproken worden die eigenlijk bij Beperkingen en Performance thuishoren.

3.2.1 HyperCard als Flat File Manager.

Men kan HyperCard gebruiken voor het uitvoeren van bepaalde operaties zoals die gevonden worden in file management software. Bijvoorbeeld kan HyperCard gebruikt worden voor het ontwerpen van informatie-invoerformulieren op het scherm en het zoeken naar een bepaalde vorm. Hier moeten echter direct belangrijke verschillen worden opgemerkt.

Met HyperCard heeft men volledige controle over de manier waarop de formulieren verschijnen. Men wordt zelfs aangemoedigd om formulieren te ontwerpen die zo veel mogelijk op niet-gecomputeriseerde objecten lijken. Met 'formulier' wordt hier niet alleen zoiets als een sollicitatie-formulier bedoeld, maar een HyperCard formulier kan virtueel alles zijn wat men op een Macintosh scherm kan creëren: een rolodex kaart, een kalender, een afspraken- boek, een agenda, etc.. Met andere woorden, men kan HyperCard gebruiken om zowel gecomputeriseerde versies van "real-life" objecten te creëren, alsook omgevingen die onmogelijk zijn in onze wereld van inkt en papier.

HyperCard wordt vaak gedefinieerd als een flat file database programma (zie boven). Zoals Danny Goodman aangeeft in [3] komt dit waarschijnlijk omdat kaarten velden bezitten zoals database-programmaschermen velden hebben voor tekst en numerieke informatie. Echter, hoe verder je het database paradigma uitrekt door te zeggen dat kaarten gelijk zijn aan database records en dat stacks hetzelfde zijn als database files, hoe verder je van HyperCard afdwaalt.

Hoewel een HyperCard kaart en een database entry er hetzelfde uitzien, bestaat er een groot verschil in de manier waarop elk systeem de informatie die in de velden wordt ingevoerd behandeld (dit verschil bepaalt mede voor welk

systeem men uiteindelijk kiest).

Elke keer als informatie in een HyperCard kaartveld wordt ingevoerd, wordt de tekst opgeslagen als data die bij de kaart hoort. M.a.w., de hele kaart wordt opgeslagen (natuurlijk worden niet de gemeenschappelijke achtergrondplaatjes en andere gemeenschappelijke attributen bij iedere kaart opgeslagen, maar wel die items die elke kaart verschillend maken, dus zijn naam, id nummer, script en tekst etc.).

In de meeste databases echter zijn er geen on-screen kaarten maar slechts een entry formaat dat gebruikt wordt als template voor informatie die ingevoerd of opgevraagd wordt. Als men hier data invoert in de velden van het template en men drukt op "return" dan wordt de data opgeslagen in de database file in een soort lijst formaat. De informatievelden van een enkele entry worden samen als een record (een rij van de lijst) bewaard.

Een belangrijk voordeel van het op deze manier opslaan van informatie is dat het makkelijk is om de data in geselecteerde lijsten op het scherm te tonen, iets dat HyperCard niet als standaard functie heeft. Oftewel, men kan een bepaalde view op de data genereren. (Het is wel mogelijk dit in HyperCard te doen maar dit kost veel moeite, meer hierover in Beperkingen).

3.2.2 HyperCard als Relationeel Database Systeem.

HyperCard beschikt ook over elementaire relationele eigenschappen. Het is mogelijk om informatie opgeslagen in de ene stack van HyperCardformulieren op te laten vragen door formulieren in andere stacks. HyperCard breidt dit relationele gedrag zelfs uit.

In tegenstelling tot een relationele database staat HyperCard het toe om echt een sprong te maken naar de andere stack om de volledige context van de gerelateerde informatie te bekijken. Dus waar een relationele database zijn relationele capaciteiten beperkt tot het simpel betrekken van informatie ergens anders vandaan laat HyperCard iemand werkelijk naar deze informatie toegaan en laat hem rond springen afhankelijk van zijn informatie-behoefte. Natuurlijk ligt hierin ook een nadeel verscholen, namelijk dat de gebruiker wordt overvallen door een stortvloed van informatie waar hij helemaal niet om gevraagd heeft en dat hij rondspringt in informatie waar hij niets te maken heeft.

HyperCard wordt mede hierdoor vaak vergeleken met relationele database systemen. Hierover merkt Danny Goodman in [3] op dat hoewel HyperCard relatie-achtige acties kan uitvoeren, het niet bedoeld is om relationele databases te vervangen.

Het volgende voorbeeld laat duidelijk het verschil tussen een relationeel database systeem en HyperCard zien. Veronderstel dat een maatschappij gegevens over klanten bijhoudt (naam, adres, telefoonnummer, kredit, etc.) in een klantendatabase. Er wordt ook een aparte database bijgehouden voor bestellingen; iedere bestelling opgegeven door een klant

wordt op een bestellingsformulier ingevuld. Het relationele aspect komt om de hoek kijken als degene die het bestelformulier invult de klantgegevens moet invoeren. In het geval van een relationele database hoeft de operator het klant-id-nummer slechts in te voeren, en door vooraf aangegeven links tussen de bestellingendatabase en de klantdatabase wordt het klant-id-nummer gebruikt om de klantgegevens op te zoeken en deze in te vullen op het bestelformulier.

Wat maakt deze operatie in een relationeel database-systeem nu anders dan in het HyperCard equivalent (HyperCard kan bovenstaande actie zonder moeite imiteren) ?

Het verschil zit in de manier waarop de informatie is opgeslagen in de files. In het geval van een relationele database worden de klantgegevens (op het id-nummer na) niet echt bij de bestellinginformatie opgeslagen. Klantgegevens blijven in de klantdatabase. Als er een formulier op het scherm verschijnt dat de klantgegevens nodig heeft dan zoekt het relationele database-systeem de gegevens op en plaatst ze (alleen) op het scherm in de daarvoor bestemde velden. In het geval van HyperCard echter zal de bestellingen-stack de gegevens uit de klantstack ophalen en ze op de juiste plaatsen op de bestelkaart plaatsen. Hier wordt de data echter bij de bestellingenkaart en -stack opgeslagen.

Afhankelijk van het ontwerp van het stack-systeem kan het opslaan van gegevens op meerdere plaatsen voordeel bieden. Bijvoorbeeld, als de gegevens in het veld staan kunnen zij beschermd worden tegen veranderingen buiten de stack, zelfs wanneer de klantstack beschadigd wordt. Geen enkele stack is nog afhankelijk van een andere voor het weergeven van veld data. Iedere stack wordt een alleenstaande databank welke overgedragen kan worden aan andere computers of gebruikt kan worden door gebruikers die geen toegang hebben tot de andere stacks (bijvoorbeeld wel toegang tot de bestellingendatabase maar niet tot de klantgegevens) van het systeem.

Het grootste nadeel is natuurlijk dat de gegevens in de besteldatabase niet automatisch veranderen als ze dat wel in de klantdatabase doen (tenzij men dit uitdrukkelijk programmeert, hetgeen weer extra overhead met zich mee brengt en dus prestatieverlies). Dit is maar één nadeel van redundante informatie. Bij een relationeel systeem zou een verandering in de klantdatabase doorwerken door het hele systeem omdat de data slechts opgeroepen wordt wanneer deze nodig is. Men kan dus altijd over de meest recente data beschikken. Hoewel ook aan de relationele aanpak nadelen kleven -zoals het verlies van betekenis en de problemen die optreden bij het invullen van het bestellingenformulier als de klantdatabase verwijderd of beschadigd is (zie mijn afstudeerverslag).

In een erg gestructureerde HyperCard-omgeving (terwijl de mogelijke ongestructureerdheid en dus grotere vrijheid juist een van de pluspunten is) zou het mogelijk zijn om de veranderingen door het hele systeem te maken. Dit kan gedaan worden door een zorgvuldig geschreven script dat actie onderneemt op het moment

dat belangrijke velden in de klantdatabase gewijzigd worden. Maar omdat HyperCard stacks vrij door de gebruiker veranderd kunnen worden (tot op zekere hoogte) kan een ontwerper er niet zeker van zijn dat de beginstructuur van de database zal overleven.

Het is (juist) de bedoeling van HyperCard om de ontwerper en gebruiker te vrijwaren van dit soort vaste structuren die de formele databases zo lang van ons verlangden. En bij de beperkingen zoals die door de stugge structuren aan de relationele database systemen worden opgelegd, ligt nu juist een van de sterke punten van HyperCard, namelijk de mogelijkheid om willekeurige links te leggen tussen welke bases van data dan ook.

Men kan HyperCard ook gebruiken om specifieke links tussen formulieren te leggen, dit om het zoekproces naar relevante informatie te versnellen. Deze links zijn niet definitief. Het is toegestaan om nieuwe links te leggen en oude te verwijderen of aan te passen.

Als men een relationeel databasemodel bouwt, moet men voorzichtig zijn met het bepalen van hoe de ene database-sectie informatie in een andere opzoekt. De manier waarop toegang tot de database wordt verkregen bepaalt vaak hoe de database is gestructureerd, zoals bijvoorbeeld of het klant-id-nummer de bepalende factor is die het ene record van het andere onderscheidt. Als men later ontdekt dat men eigenlijk een andere manier van toegang krijgen wil, raakt men bij relationele database systemen vaak in de problemen.

In HyperCard echter zijn er geen structuren die bepalen hoe een stack of kaart georganiseerd moet zijn. Men mag data van een stack onttrekken door te zoeken naar tekst in een willekeurig veld; men mag data die verkregen is door selectie met elk willekeurig en gewenst criteria toevoegen aan een specifieke kaart in een andere stack, zelfs als de link naar die stack via andere stacks en links loopt.

Tot zo ver de vergelijking van HyperCard en de database categorieën. In de volgende paragraaf zullen nog enkele algemene eigenschappen van databases en hoe HyperCard hieraan voldoet besproken worden.

3.3 HyperCard en database eigenschappen.

Een groot verschil tussen HyperCard en DBMS software is de manier waarop gevonden informatie zichtbaar wordt gemaakt en wordt geprint. Een belangrijke eigenschap van DBMS is dat ze rapporten kunnen genereren op vele manieren. HyperCard is echter niet ontworpen om rapporten te genereren, hoewel er wel zoiets mogelijk is. In plaats daarvan is HyperCard geoptimaliseerd om op zoek naar de gewenste informatie snel door de bestaande kaarten te bladeren. Een DBMS programma kan beter een rapport produceren gebaseerd op de gewenste selectiecriteria (aangenomen dat men het formaat van het rapport al heeft gespecificeerd) en HyperCard kan beter de kaarten vinden (en tonen) die aan de criteria voldoen.

Databases zijn over het algemeen lijst-georiënteerd. HyperCard is daarentegen een "browsing" omgeving waarin de data het best tot zijn recht komt in on-screen kaarten en waarin het makkelijk is om toegang te krijgen tot deze kaarten in zowel een lineaire, sequentiële volgorde als in een niet-lineaire, niet-sequentiële, "spring-rond" volgorde. Vreemd genoeg kan eenzelfde hoeveelheid informatie geschikt zijn om zowel in HyperCard als in de traditionele databases op te slaan; niet door zijn inhoud maar door de manier waarop de gebruiker moet werken met die informatie. Men zou kunnen overwegen om een database in HyperCard te implementeren als men slechts door de informatie wil bladeren of als men iets snel op wil zoeken (met het snelle HyperTalk Find-commando). Ook als men een actie gerelateerd aan bepaalde informatie wil ondernemen zoals het bellen van klanten uit het klantbestand kan men dat eenvoudig in HyperCard doen. Ook is het mogelijk om ingewikkelde, maar daardoor langzamere zoektechnieken te gebruiken om een bepaalde selectie kaarten te bekijken. Een traditionele database moet gebruikt worden als de gebruiker een view van bepaalde selecties nodig heeft, in verschillende on-screen en geprinte rapporten.

Het grootste verschil is dat een database programma de gebruiker in staat stelt om meervoudige selecties (bijvoorbeeld postcode tussen 2000 en 3000 EN achternaam beginnen met A of M) op de hele database toe te passen. Het selecteren volgens zoek-criteria zoals bovenstaand lijkt op het maskeren van alle andere data in de files. Waar een database veel beter in is, is dus vooral het genereren van on-screen en geprinte rapporten van de geselecteerde data, hoewel HyperCard deze achterstand snel inloopt door add-on producten als Reports (en volgens mij is het maskeren van data en het uitvoeren van meervoudige selecties ook niet al te moeilijk).

In boeken wordt gegeven dat voor een database, het ontwerp en gebruik, nodig zijn:
- een data definition language (DDL);
- een data manipulation language (DML);

- een data control language (DCL);
- een query language;
- DBM software tbv de database administrator.

HyperCard voorziet hier ook in, zij het dat het niet allemaal aparte formele talen zijn. Alles is in principe mogelijk in HyperTalk (zie ook hoofdstuk 7).

Uitgesplitst ziet het er als volgt uit:

*) HyperCard voorziet in makkelijke stappen om een database te ontwerpen en op te bouwen, dit is het knip- en plakwerk met behulp van menu's.

*) Verder is reorganiseren en het veranderen van data geen probleem. Over het algemeen gebeurt ook dit interactief door de gebruiker met behulp van menu's.

*) HyperCard voorziet ook in elementaire toegangscontrole- structuren welke in HyperTalk uitgebreid kunnen worden (hier ben ik momenteel mee bezig).

*) HyperCard (en de Macintosh) zijn zoals gezegd erg gebruikersvriendelijk. Dus is het niet moeilijk om een eenvoudige, gebruikersvriendelijke interface en query taal te ontwerpen. Ook beschikt HyperCard al over het elementaire commando Find.

*) De DBM software zijn de scripts waarmee HyperCard en de database beschreven zijn. Deze kunnen afgeschermd worden voor de gebruiker.

3.4 HyperCard of traditionele database ?

Danny Goodman geeft in [3] vier vragen aan de hand waarvan kan worden bepaald of de HyperCardomgeving het beste alternatief is om de applicatie in te implementeren:

- Kan de informatie van de applicatie opgesplitst worden in kaarten ter grootte van een Macintosh scherm zonder onoverzichtelijk te worden ?
- Is de wens om uitgebreide rapporten te maken beperkt of in ieder geval van die aard dat HyperCard add-ons eraan kunnen voldoen ?
- Wil men dat de applicatie grappig en uitnodigend om te gebruiken is, speciaal voor mensen die niet met computers of niet met Macintosh gewerkt hebben ?
- Is het geoorloofd dat de gebruiker de applicatie aan kan passen ?

Als het antwoord op een van deze vragen 'Ja' is en de andere vragen worden niet met 'Nee' beantwoord dan kan HyperCard de goede keus zijn, volgens Danny Goodman.

Danny Goodman merkt ook in [3] op dat als de database ontworpen is om de gebruiker te voorzien van een lijst on-screen van uit de hele database geselecteerde informatie, het niet veel zin heeft om de database naar HyperCard om te zetten. Ook als de database sterk relationeel is, hetgeen betekent dat het zwaar afhankelijk is van data gegenereerd uit vele andere

dataverzamelingen, is het vanuit prestatie-oogpunt niet verstandig om de database om te zetten naar een HyperCard applicatie.

Als een database daarentegen voornamelijk read-only is en sterk encyclopedie-achtig is het zeker de moeite waard om te overwegen de applicatie in HyperCard te maken.

Als de vraag of de applicatie wel geschikt is om in HyperCard te representeren bevestigend is beantwoord, zullen de volgende vragen nog beantwoord moeten worden:
- hoe moeten de velden herordend worden ?
- hoe kan de rapportage worden beïnvloed ?
- wat te doen met lange "chunks" tekst ?

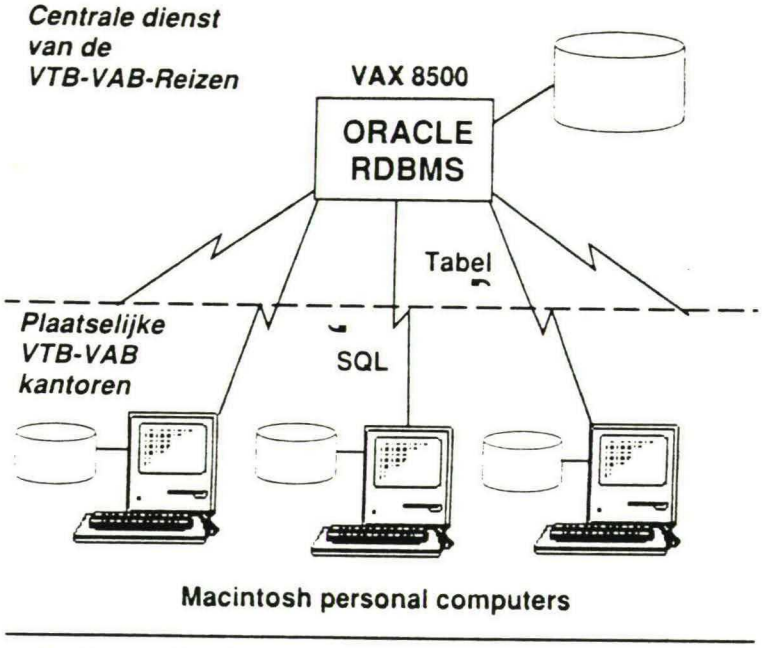
Hier stopt de beschrijving van HyperCard en Databases. In het volgende hoofdstuk zal behandeld worden hoe HyperCard gebruikt is samen met een bestaande database.

Hoofdstuk 4.

HyperCard in bestaande database systemen.

HyperCard kan niet alleen, zoals hierboven is beschreven, zelf dienst doen als database, maar zeker zo goed, zo niet beter, als front-end voor een bestaand database- systeem. Het is hier zeer geschikt voor vanwege de uiterst ver doorgevoerde gebruikersvriendelijkheid van zowel HyperCard als de Macintosh.

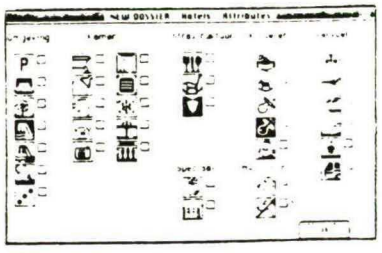
Een erg goed voorbeeld in dit geval is het CATS-systeem van de VTB-VAB Reizen (de Belgische ANWB) (zie figuur 2).



Er zijn zowel locale als centrale tabellen. De statische gegevens (bvb. inlichtingen over accomodaties van Hotel X) bevinden zich locaal; de dynamische gegevens (bvb. inlichtingen over de stand van reservaties in hotel X) bevinden zich centraal. De verkoper stelt met behulp van de Macintosh computer vragen over beide soorten tabellen zonder dat hij hoeft te weten waar deze tabellen zich bevinden.

figuur 2. CATS-systeem.

In het CATS-systeem dienen Macintosh-computers met HyperCard als front-end voor een ORACLE RDBMS op een VAX 8500 computer. Met behulp van HyperCard kan de verkoper samen met de klant de meest geschikte reis en accomodatie voor de klant vinden door vragen te beantwoorden in allerlei categorieën die gerepresenteerd worden door icons en dialogen, zie figuur 3.



figuur 3. CATS-interface.

Het is een snelle methode (er is een rechtstreekse verbinding met de centrale computer) en door het gebruik van HyperCard en de gebruikersvriendelijke interface hoeft de verkoper geen weet te hebben waar precies de informatie zich bevindt en hoe deze is opgeslagen, ook hoeft hij zelf geen 'queries' in elkaar te draaien. Het enige dat de verkoper hoeft te weten is hoe hij door het klikken op icons en het beantwoorden van vragen de meest geschikte keuze samen met de klant kan maken.

Het kiezen van icons en het beantwoorden van vragen wordt in HyperCard omgezet naar een of meerdere SQL-queries die via een telefoonverbinding naar de centrale computer gestuurd worden. Het eventuele antwoord wordt dan weer op een grafisch aantrekkelijke manier aan verkoper en klant getoond.

Bovenstaand voorbeeld geeft aan hoe HyperCard als interface voor bestaande databases kan dienen (door het aanmaken van queries op een gebruikersvriendelijke manier zodat de gebruiker niets van queries of opslag zelf hoeft af te weten). Ook geeft dit voorbeeld aan dat HyperCard eigenlijk als deel van een soort gedistribueerde database wordt gebruikt. De statische gegevens van accomodaties worden in HyperCard zelf opgeslagen en de dynamische gegevens over reizen (zoals vluchtschema's, reserveringen, etc.) worden centraal opgeslagen.

Een ander voorbeeld is QFE (Query Farandole Easily). Farandole is net als in bovenstaand voorbeeld een database systeem op een VAX/VMS en gebruikt HyperCard als interface. QFE gaat zelfs verder dan bovenstaand voorbeeld, want in QFE is de interface op een hypertext-manier georganiseerd en dit hypertext systeem is in HyperCard geïmplementeerd. Ik geef hieronder een citaat uit [10] dat het gebruik van HyperCard in QFE beschrijft:

"For each database in order to use QFE on it, a network of cards has to be created. This network is an instance of the query interface over the database. The database designer uses a specific tool to interactively build the graph and compose the schema. This tool is called MFE (Model Farandole Easily) and was also built with HyperCard. The designer defines by that way the relations, attributes, domains and contexts of this database. All the information is stored in the data dictionary. Once the

dictionary is complete, a process can be activated to automatically generate an instance of the interface for this database, i.e. a specific network of cards. This process starts from an empty skeleton of the interface and fills it with the contents of the data dictionary. It creates cards and links between cards that permit navigation through the interface".

Hoofdstuk 5.
Hoe kan HyperCard gebruikt worden als een (Intelligent) Database Systeem ?

Dit hoofdstuk tracht een antwoord te geven op de vraag of HyperCard, gebruikmakend van zijn vermogen om ongelimiteerd links te leggen tussen verschillende soorten informatie, ook gebruikt kan worden als intelligent soort database. Voordat ik dat kan doen, moet ik eerst een antwoord geven op de vraag wat intelligente database systemen zijn.

5.1 Intelligente database systemen.

Bovenstaande vraag is een ingewikkeld probleem waar veel mensen al jaren een duidelijk antwoord op proberen te geven.

Ikzelf zou de volgende omschrijving van een intelligent database-systeem willen geven:

Een intelligent database-systeem bestaat uit een intelligente interface en een intelligente database.

5.1.1 Intelligente Interface.

Een intelligente interface is een gebruikersvriendelijke interface die aan de hand van een dialoog met de gebruiker, waarin ook rekening gehouden wordt met fouten die de gebruiker kan maken, een geoptimaliseerde query op de database doet. De dialoog wordt gestart door het opgeven van de initiële query door de gebruiker. Een voorbeeld hiervan zou zijn dat de gebruiker vraagt naar uitgaansmogelijkheden in 'Amstredam' welke vraag toch goed geïnterpreteerd wordt. Ook wordt in de dialoog getracht de query zo optimaal en precies mogeljk te maken. Men zou dit een soort semantische queryoptimalisatie kunnen noemen omdat naar de bedoeling van de gebruiker wordt gezocht.

Een voorbeeld zou zijn dat als de gebruiker vraagt naar gegevens over het gelukkig-zijn van zijn klanten, de dialoog de query uitbreidt naar informatie over blije klanten, klanten die niet droevig of boos zijn, klanten die "mazzel" hebben, en de redenen voor hun gelukkig-zijn. Een ander voorbeeld is het geval dat een gebruiker de database wil ondervragen over attracties, waarna door de dialoog de query uiteindelijk zo is beperkt dat alleen naar musea voor moderne kunst in Amsterdam gevraagd wordt. Het systeem beperkt natuurlijk niet naar willekeur, maar selecteert op grond van criteria die voor die bepaalde query, voor die bepaalde gebruiker, relevant zijn, bijvoorbeeld omdat die gebruiker fervent museumbezoeker is, en het systeem daar weet van heeft. Ook wordt deze intelligente interface gebruikt voor een begrijpelijke en door de gebruiker gewenste output van de query.

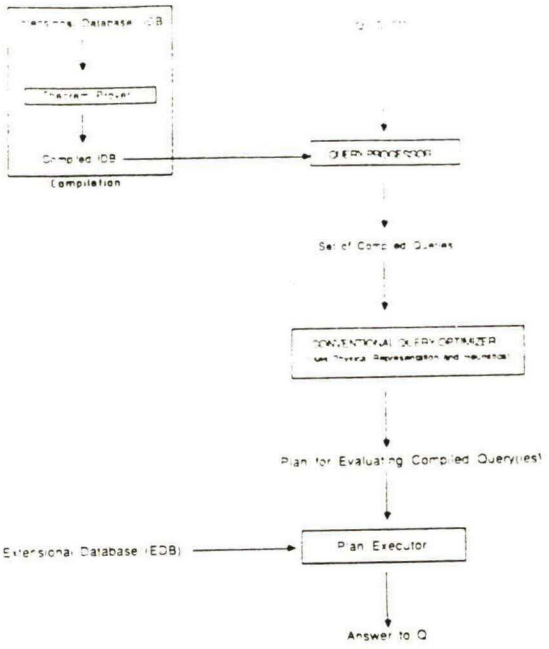
Een intelligente interface is verder een interface die het schema van de database aan de gebruiker duidelijk weet te maken zodat deze een begrip krijgt van de door de ontwerpers ingebouwde semantiek. Tot slot moet de interface de gebruiker helpen bij het zoeken naar de juiste identifiers zodat de gebruiker deze niet allemaal hoeft te onthouden.

5.1.2 Intelligente database.

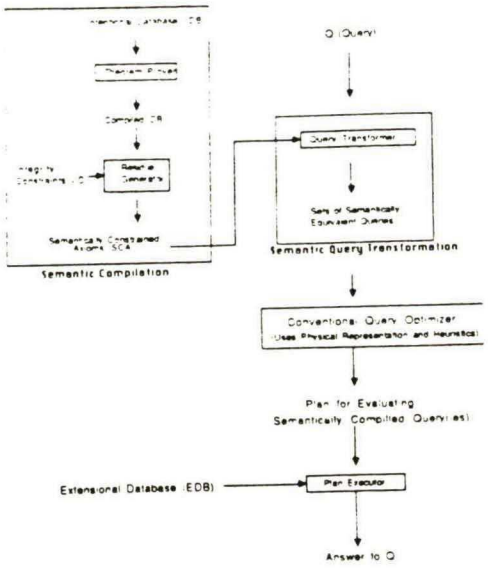
Gezien mijn achtergrond (afstudeerwerk op het gebied van deductieve semantische database systemen) is het niet vreemd dat ik een intelligente database omschrijf als een database waar naast feiten en relaties uit een te representeren wereld ook de betekenis van de relaties tussen de feiten, hoe de relaties tot elkaar staan, en constraints over de feiten en relaties een rol spelen, oftewel een database waar ook de semantiek van de opgeslagen gegevens een rol speelt. Verder moet de database daardoor in staat zijn om dynamisch nieuwe, semantisch relevante gegevens af te leiden uit de al in de database aanwezige kennis.

Tevens moet een database om intelligent gedrag te kunnen vertonen kennis hebben over zichzelf en zijn inhoud en hoe ermee te werken, oftewel er moet meta-kennis kunnen worden opgeslagen (en afgeleid).

Ook in dit deel komt query-optimalisatie om de hoek kijken. Ook hier gaat het om semantische query- optimalisatie, hetgeen betekent dat door de in de database aanwezige kennis de query zo wordt beperkt dat alleen naar relevante informatie gezocht wordt. Voor zo'n intelligente database grijp ik terug op mijn afstudeerscriptie [11] en het werk van Jack Minker, die zich bezig heeft gehouden met semantische databases. Tevens verwijs ik naar een artikel van Chakravarthy, Grant en Minker in [6] waarin Minker's database wordt uitgebreid met een semantische query optimalisatie. Zie voor beide onderdelen onderstaande figuren:



figuur 4.
Semantisch deductieve database



figuur 5.
Semantisch deductieve database met semantische query-optimalisatie.

5.2 De object-georiënteerdheid van HyperCard.

Hoe HyperCard als database systeem te gebruiken is, is uitvoerig beschreven in hoofdstuk 3 'HyperCard en database systemen'. In dat hoofdstuk en in [1],[2],[3] wordt alleen aandacht geschonken aan de manieren om flat file managers of relationele database-systemen (met tabellen en lijsten) in HyperCard te implementeren. Tevens wordt er melding gemaakt van de object-georiënteerde eigenschappen van HyperCard. Er wordt echter geheel voorbijgegaan aan de rol die deze eigenschappen kunnen spelen bij een heel ander soort database-systeem.

HyperTalk, de programmeertaal van HyperCard, wordt object-georiënteerd genoemd en HyperCard zelf wordt gedefinieerd door objecten en de message handlers voor de messages die deze objecten elkaar zenden. Het zou dus ook mogelijk moeten zijn om HyperCard te beschouwen als soort van object-georiënteerde database. Dat hier ook nog wat haken en ogen aanzitten zal hieronder blijken. Voordat ik hier verder op inga, zal ik eerst nog beschrijven hoe object-georiënteerd HyperCard en HyperTalk zijn.

In hoofdstuk 1 'Wat is HyperCard' zijn de verschillende objecten (stack, background, card, field en button) waaruit HyperCard bestaat al beschreven. En zoals hierna wordt gegeven zijn HyperTalk scripts (programma's, een ander element van HyperCard) gebeurtenis-georiënteerd zijn. Omdat de scripts zelf

zijn HyperTalk scripts (programma's, een ander element van HyperCard) gebeurtenis-georiënteerd zijn. Omdat de scripts zelf ook als samenwerkende objecten beschouwd kunnen worden, en zij communiceren via boodschappen (messages) is er geen begin of eind van een script aan te geven.

HyperTalk acties worden ondernomen aan de hand van boodschappen die zij ontvangen. Deze boodschappen worden gegenereerd als er bepaalde gebeurtenissen plaatsvinden. Men noemt HyperTalk daarom gebeurtenis-georiënteerd (event oriented). De messages die de bepaalde executie van een script triggeren kunnen systeem-gegenereerde boodschappen zijn (zoals idle, resume of quit), maar ook een door de gebruiker gegenereerde boodschap (zoals het drukken op de mouse-knop of het binnengaan van een veld). Ook scripts zelf kunnen boodschappen naar andere scripts sturen. De boodschappen volgen een bepaald pad in de HyperCard organisatie. Als een message gegenereerd wordt, zoekt hij in bepaalde objecten naar een bijpassende message handler. Als een object de message ontvangt en er geen handler voor heeft dan zendt het object de message door naar het volgende object in de HyperCard-hiërarchie. Het volgende plaatje geeft deze hiërarchie aan:

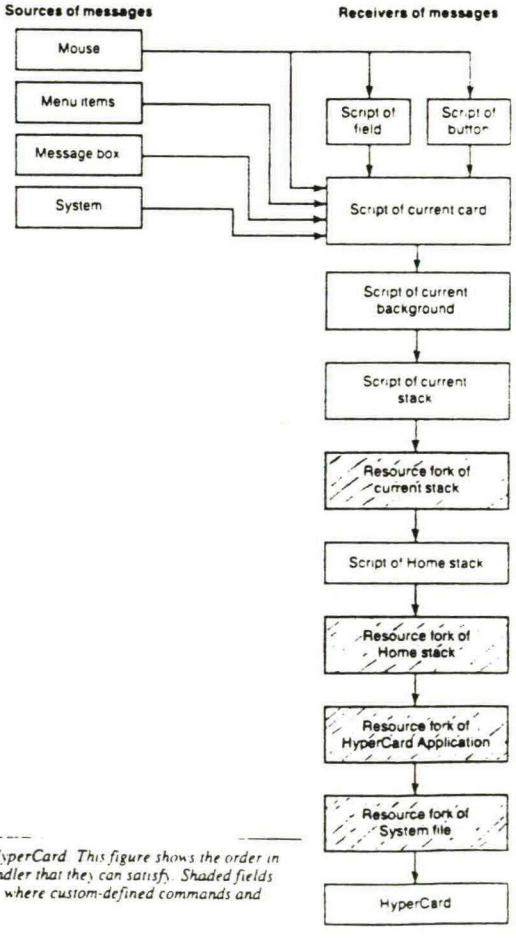


Figure 6: *Inheritance hierarchy within HyperCard. This figure shows the order in which messages search for a message handler that they can satisfy. Shaded fields denote the resource forks of certain files, where custom-defined commands and functions may reside.*

figuur 6. De HyperCard objecthiërarchie.

Deze elementen (objecten, messages en handlers) maken van HyperTalk een simpele object-georiënteerde programmeertaal (OOT). Het opereert op verschillende soorten objecten die elkaar boodschappen sturen. De message-handlers kunnen vergeleken worden met methoden zoals die in andere OOTs voorkomen. De hiërarchie maakt het voor sommige objecten mogelijk om bepaalde message-handlers te erven van andere objecten of ze te overschrijven met hun eigen handlers, en cards erven objecten als velden en buttons plus hun scripts van backgrounds.

De haken en ogen waarover eerder gesproken werd, is dat in vergelijking met "echte" object-georiënteerde talen HyperCard de mogelijkheid mist om nieuwe klassen te creëren, en ook is de HyperCard hiërarchie geen echte OOT-hiërarchie. Een laatste punt is dat de locale informatie van een HyperCard-object wel door andere objecten te bereiken is, er is dus geen "data encapsulation". Deze problemen maken dat HyperCard niet zondermeer als een object-georiënteerde database dienst kan doen maar dat er verschillende aanpassingen nodig zijn.

5.3 Kennisrepresentatie in HyperCard.

Ik heb op dit moment slechts weinig gestructureerde ideeen over hoe precies deze aanpassingen eruit moeten zien om HyperCard geschikt te maken voor kennisrepresentatie in de objecten en -hiërarchie. Ze zijn nog te vaag om aan dit papier toe te vertrouwen. Natuurlijk kan ik al wel zeggen dat kennis (wat is kennis ?) wordt opgeslagen in de objecten van HyperCard en in de handlers die bij de objecten behoren. Ook is kennis bevat in de verschillende links die er tussen objecten bestaan.

De vraag of HyperCard gebruikt kan worden voor het ontwikkelen van een intelligente database kan door mij dus (nog) niet beantwoord worden. Zoals HyperCard er nu uitziet zijn er wel enige mogelijkheden op het gebied van de interface en het afleiden van nieuwe relaties (waarvan ik in het volgende hoofdstuk een simpel voorbeeld geef), maar ik voorzie grote problemen bij het invoeren en bewaken van constraints en het opslaan van, en werken met meta-kennis.

De belangrijkste vraag die beantwoord moet worden om van enig succes verzekerd te zijn is:

Zijn de objecten waaruit HyperCard nu bestaat voldoende om er een redelijk complex, intelligent database systeem mee te bouwen of breekt de onmogelijkheid om nieuwe classes te definiëren op ?

Het volgende hoofdstuk geeft aan hoe in HyperCard de huidige objecten gebruikt kunnen worden om een, nog niet intelligente, database te implementeren.

Hoofdstuk 6.
Een database van HyperCard objecten.

In dit hoofdstuk wordt de object-georiënteerdheid van HyperCard, zoals beschreven in het vorige hoofdstuk, gebruikt om databases te beschrijven.

6.1 Analogie tussen database en HyperCard-objecten.

Naast het representeren van een database als een aantal lijsten of tabellen van gegevens kan men een database ook bekijken als een aantal objecten met onderlinge relaties. Deze laatste kijk op databases voldoet aan het beeld dat van HyperCard gegeven wordt.

Een analogie tussen databases en HyperCard-objecten zou de volgende kunnen zijn: Een stack is een database, een background staat voor een class en een card is een object uit een class. De velden van een card zijn de attributen van het object en de buttons en bijbehorende links geven de relatie met andere objecten (uit dezelfde of een andere klasse) aan. Alle objecten uit een class bezitten de gemeenschappelijke attributen of eigenschappen van die class. Eigenschappen van relaties, attributen, object en class kunnen beschreven worden in de handlers van de desbetreffende objecten. Het is tevens mogelijk om in deze handlers regels te beschrijven. Regels die aangeven wat er moet gebeuren als het object gemanipuleerd wordt, of hoe het object zich in een relatie gedraagt. Hiermee wordt ook de mogelijkheid geboden om afleidingsregels op te nemen. Naast regels kunnen ook (domein-) constraints in de handlers gespecificeerd worden. Buttons (links) kunnen als relatie dienst doen. Niet alleen omdat de buttons fysiek met de muis moeten worden ingedrukt maar ook in scripts "ingedrukt" kunnen worden en daarmee de links afgelopen kunnen worden. Ook velden kunnen een relatie weergeven (zie voorbeeld).

Het vervolg van deze gedachtengang is dat relaties (buttons en eventueel velden) ook als attribuut van een object gezien worden, wat overeenstemt met hetgeen ik in mijn afstudeerscriptie geschreven heb.

Andere manieren om naar de objecten van HyperCard te kijken zijn:

De stack als relatie met de background, met buttons en velden als attributen en de kaarten als de elementen (tuples) van de relatie,

of,

de background geeft de relatie met de attributen en de kaarten zijn de tuples. De stack is dan een database.

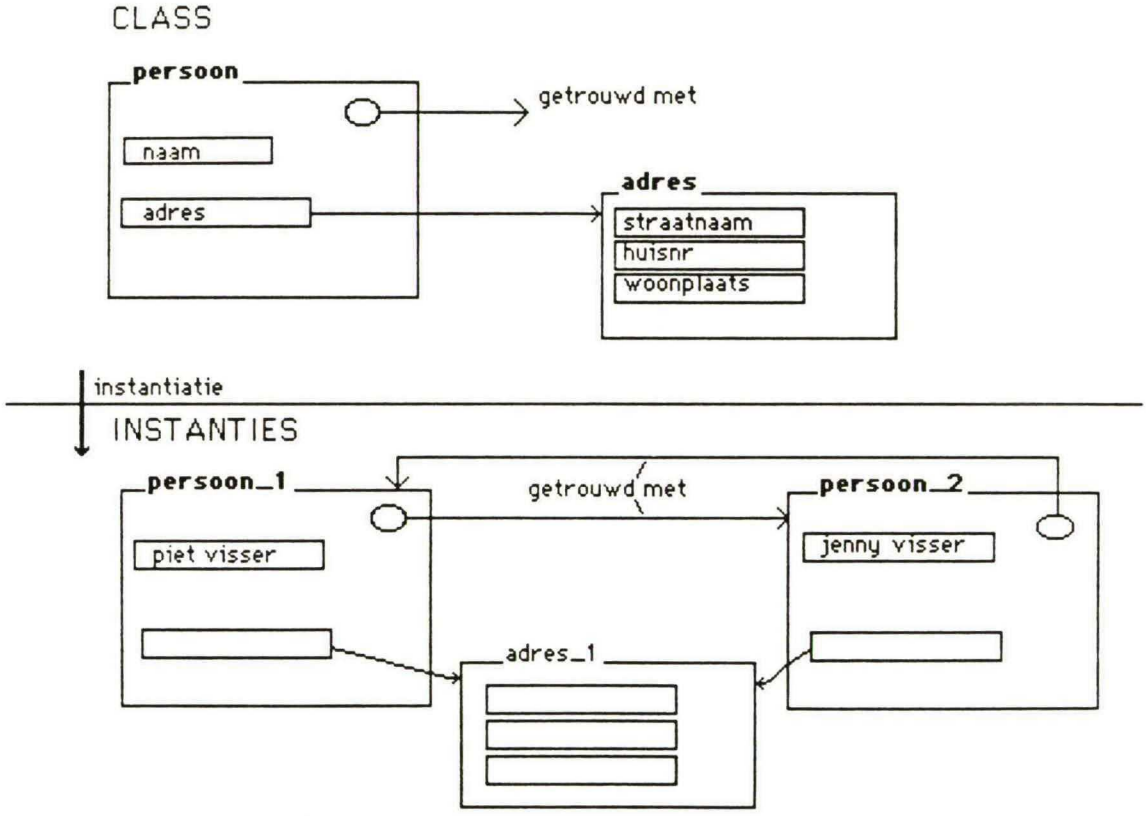
Welke van bovenstaande zienswijzen de meeste mogelijkheden geeft moet nog nader worden bestudeerd.

Hoe kan deze zienswijze nu gebruikt worden om een intelligente database te bouwen ?

Zoals uit bovenstaande ideeën blijkt, kunnen objecten, relaties tussen objecten, de klasse van objecten, regels om nieuwe relaties af te leiden, en ook constraints in HyperCard gerepresenteerd worden en daarmee zou de weg openstaan voor intelligente databases.

6.2 Voorbeelden.

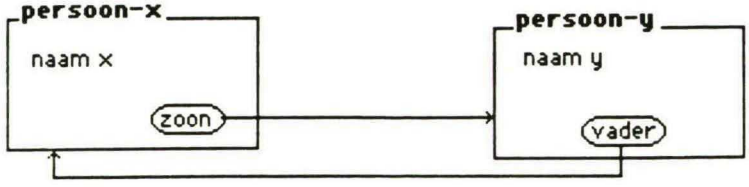
Een voorbeeld van bovenstaand eerste idee is:



figuur 7. Een database van HyperCard-objecten.

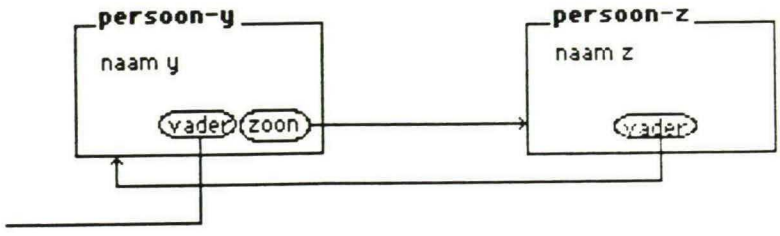
Een voorbeeld van hoe nieuwe gegevens afgeleid kunnen worden is het volgende:

Stel gegeven de volgende kaarten:



figuur 8. Voorbeeld afleiding 1.

Als persoon y nu een zoon krijgt, hetgeen als volgt in kaarten uitgedrukt wordt:

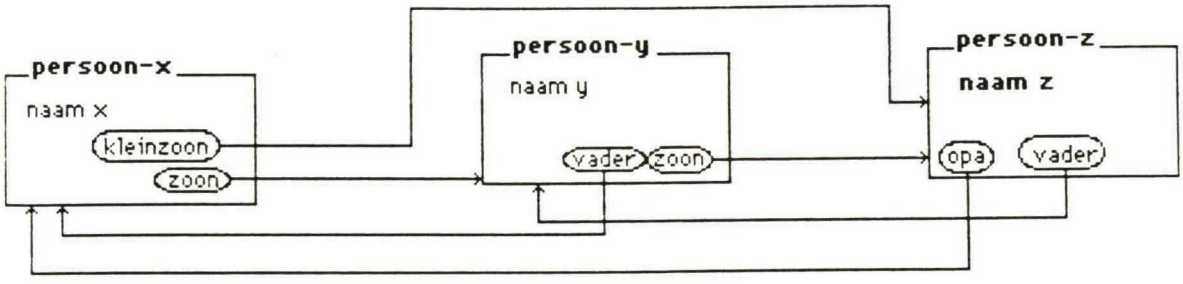


figuur 9. Voorbeeld afleiding 2.

dan kunnen als de zoon-button-handler het volgende script (in pseudo-taal) bevat de nieuwe relaties opa en kleinzoon gecreëerd worden:

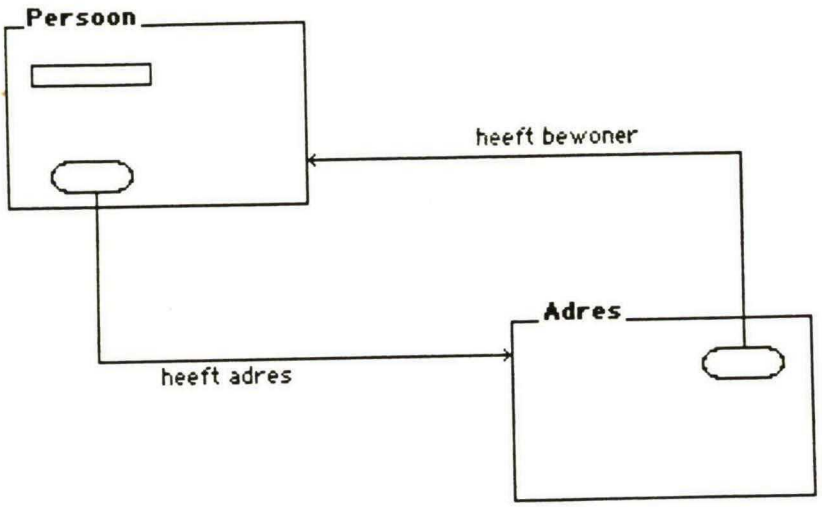
```
if newbutton(myself)
then go to zoon-kaart
     create opa-button
     link to vader-kaart
     go to vader-kaart
     create kleinzoon-button
     link to zoon-kaart
end if
```

Het resultaat van deze actie wordt dan:



figuur 10. Voorbeeld afleiding 3.

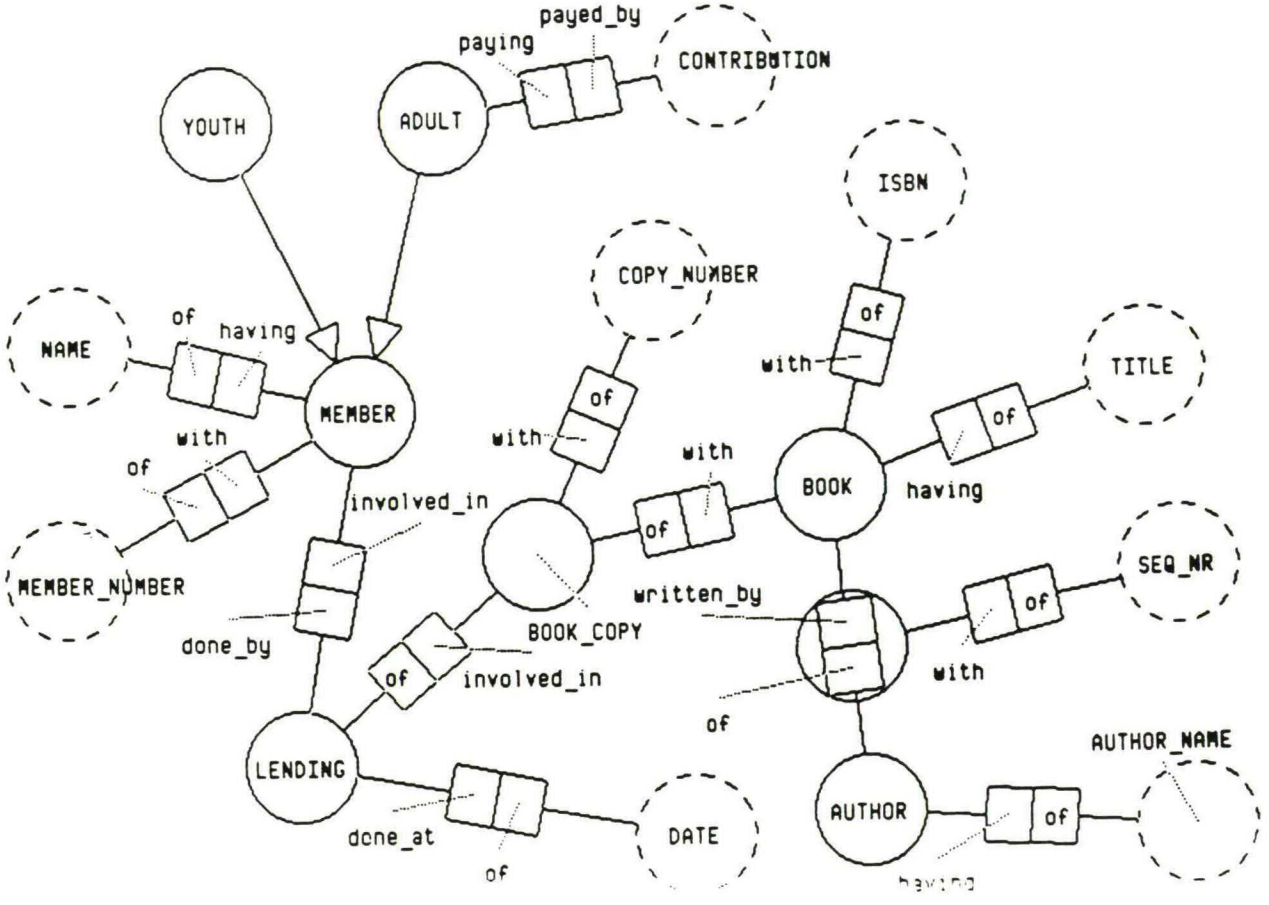
Met het idee uit de vorige paragraaf zijn ook concepten uit het Binary Relation Model te representeren:



figuur 11. NOLOTs met Fact uit BRM.

Het volgende voorbeeld zet een NIAM schema om in een equivalent van een relationele database geïmplementeerd in HyperCard. Ik heb dit gedaan om enig begrip te krijgen van hoe NIAM schema's omgezet worden en hoe een relationele database in HyperCard eruit ziet.

Hieronder wordt het ontwerp van (een deel van) een bibliotheek- informatiesysteem zoals dat in NIAM beschreven staat op blz. 14 van 'Towards models for practical reasoning about conceptual database design', van prof. Meersman [7], gegeven:

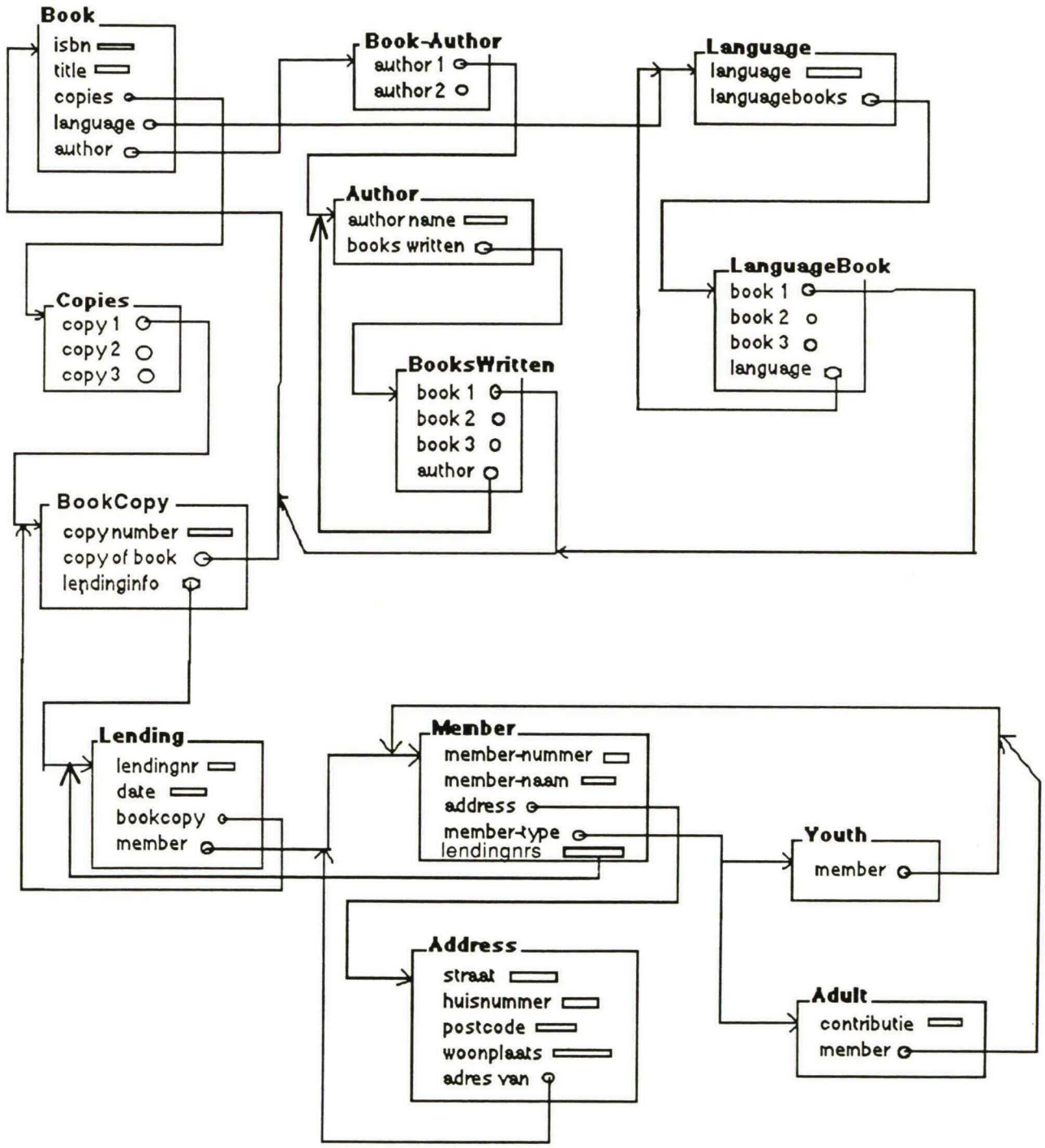


figuur 12. NIAM-schema van een bibliotheek-informatiesysteem

Bovenstaand voorbeeld zou er in HyperCard als volgt uit kunnen zien:

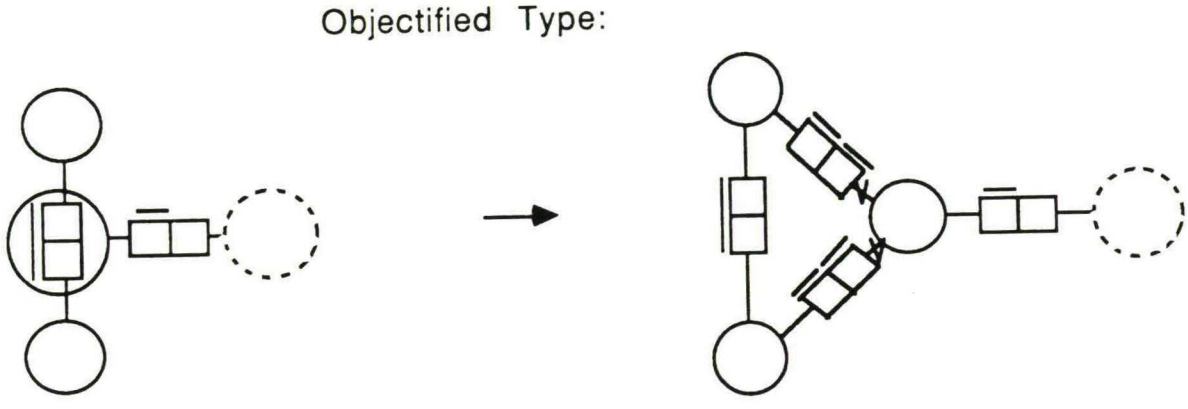
De dik gedrukte namen geven Backgrounds aan (als klasse). Bij het invullen van data zal iedere kaart (instantie) eigen buttons moeten krijgen die wel aan het concept/model voldoen, maar eigen invullingen/links naar andere instanties (van de juiste background) krijgen. De buttons (en handlers) van de background (klasse) worden dus 'overruled' door de buttons en handlers van de kaart. De velden horen zowel tot kaart als background.

figuur 13. Bibliotheek-informatiesysteem in HyperCard.



Het omzetten/mappen van het NIAM-schema in/op HyperCard objecten is als volgt gebeurd:

- elke NOLOT is een background;
- elke LOT dat via een facttype verbonden is met een NOLOT wordt een field op die background;
- het bijbehorende facttype wordt dus opgevat als: 'bezit'/'is deel van';
- elke role van een facttype tussen NOLOTs wordt afgescheiden en komt als een button terug op die background (bij inheritance moeten deze buttons gecopiëerd worden op iedere instantie (kaart));
- het objectified type wordt uitgesplitst volgens onderstaand schema waarna het op bovenstaande wijze verwerkt kan worden:

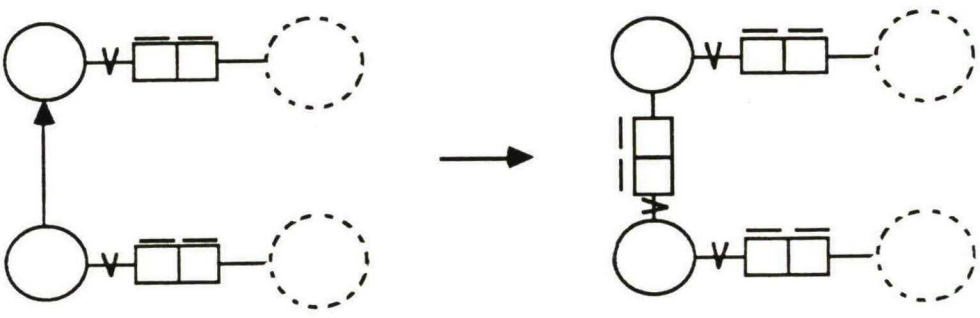


figuur 14. Objectified type.

- Het subtype komt tot uitdrukking door afhankelijk van bekende waarden voor één subtype te kiezen (of een rijtje buttons, waarvan er maar één gekozen kan worden, in de card/background op te nemen). De inheritance wordt gegeven door een link terug naar het supertype te leggen. In mijn HyperCard-versie wordt het eigenlijk ook als attribuut beschouwd.

Een andere manier is om het subtype om te zetten in onderstaand schema, met de toevoeging van semantische constraints die aangeven dat het om een subtype gaat en dat dit met zich meebrengt dat we meer over deze NOLOT kunnen zeggen dan bij normale NOLOTs:

Subtype:



figuur 15 Subtype.

- constraints:
    + identifiers die voorkomen bij een facttype tussen LOT en NOLOT kunnen gecontroleerd worden in de scripts van de background (of field);
    + N:1 of 1:N -identifiers tussen NOLOTs worden weergegeven door een extra background met verschillende buttons (het kan ook als extra buttons op oorspronkelijke background of veld dat nummers van copiën bevat (+sticky buttons));
    + de uniqueness-, total-, exclusion-constraints moeten afgevangen worden in de scripts van de betreffende velden en buttons en backgrounds. Bijvoorbeeld als een nieuwe copie van een boek in het systeem wordt ingevoerd moet, voor het copienummer is ingevoerd, gecontroleerd worden of het boek (met bijbehorend ISBN nummer) wel in het systeem voorkomt (vanwege de uniqueness-constraint).

In het voorbeeld komen geen total-roles voor. Deze zouden echter net als de andere constraints in de scripts gecontroleerd kunnen worden.

Na het omzetten van het NIAM-schema in een genormaliseerd relationeel schema is veel informatie over relaties tussen objecten, subtypes, constraints en dergelijke verloren gegaan. Deze informatie is in HyperCard veelal nog wel expliciet aanwezig. Alleen de constraints zijn niet expliciet aanwezig, maar wel impliciet in de scripts.

Hoofdstuk 7.
HyperTalk.

Zoals uit het citaat boven de samenvatting en uit het eerste hoofdstuk blijkt is HyperCard moeilijk te karakteriseren. Dit geldt eigenlijk voor de programmeertaal waarin HyperCard geschreven is, HyperTalk. Het is makkelijker om aan te geven wat je niet met HyperTalk kan doen, dan wat je er wel mee kan doen.

In paragraaf 5.2 is ook al over de object-georiënteerdheid van HyperTalk, de event-georiënteerdheid van de scripts en de hiërarchie van message handlers en objecten gesproken. Hier gaan we daar verder op in.

HyperTalk is een simpele, elegante, en erg krachtige taal, die veel op gewoon Engels lijkt. HyperTalk heeft een flexibele syntax. De zinnen lijken sterk op normale Engelse zinnen en het is dan ook in HyperTalk mogelijk om bijvoorbeeld het woordje 'the' toe te voegen of een extra spatie in te lassen. Ook vindt HyperTalk het goed dat men zowel "Go to the first card" als "Go to card 1" gebruikt en hetzelfde bedoelt. HyperTalk-zinnen bestaan uit zelfstandige voornaamwoorden (variabelen) en werkwoorden (commando's). Doordat HyperTalk zin voor zin geïnterpreteerd wordt is het ook mogelijk om een losse zin in de message box te typen die dan direct wordt geïnterpreteerd.

HyperTalk bevat veel standaardcommando's (46), controle-structuren (6), functies (49), rechtstreeks te beïnvloeden eigenschappen van de taal, HyperCard en de Macintosh (58), speciale constanten (11) en operatoren (21). HyperTalk geeft de mogelijkheid voor het declareren van procedures en functies met benoemde parameters en lokale en globale variabelen. Variabelen worden gecreëerd op het moment dat ze gebruikt worden. Als ze niet als "global" gedefinieerd zijn (overal waar ze gebruikt moeten worden), is hun scope automatisch de handler waar ze gebruikt worden. Alle variabelen worden als strings opgeslagen en als strings of nummers geïnterpreteerd (waarbij bij de laatste dan de strings worden geconverteerd).

Het is niet mogelijk om met HyperTalk het laagste hardware niveau van de Macintosh te bewerken, met andere woorden: het is niet mogelijk om rechtstreeks geheugencellen te lezen of te beschrijven. Dit is natuurlijk wel mogelijk met XCMD's of XFCN's (de externe commando's of functies, zie hoofdstuk 8). In HyperTalk zijn veel trade-offs gemaakt in het voordeel van begrijpelijkheid en gebruiksgemak en ten nadele van compactheid en snelheid.

De kracht van HyperTalk ligt onder andere in het feit dat vanuit HyperTalk scripts andere scripts (of het script zelf) te wijzigen zijn. Dus als het systeem nieuwe feiten zou leren, zouden afhankelijk van de opgegeven regels de bijpassende scripts geschreven kunnen worden of eraan gerelateerde scripts

aangepast kunnen worden.

Het is moeilijk om HyperTalk los van HyperCard te bespreken. HyperTalk is niet portable, het werkt alleen in een HyperCard omgeving. HyperTalk is de taal van alleen HyperCard en werkt op HyperCard-objecten en op de informatie die zij bevatten en nergens anders op.

Hoofdstuk 8.
De externe functionaliteit van HyperCard.

Naast de vele ingebouwde commando's en functies van HyperTalk die HyperCard al krachtig maken, biedt het ook de mogelijkheid om nieuwe commando's en functies op te nemen. Men schrijft de code voor het nieuwe commando of functie in C, PASCAL, of assembler (deze code moet wel met programma's geschreven en vertaald worden die op de Mac draaien), men vertaalt dat zonder header als type XCMD (voor commando's) of XFCN (voor functies), en plaatst dit in de resource fork van een stack (zie 'hiërarchie'). Niet alleen kunnen op deze manier alle functies van de Macintosh benut worden, maar ook kunnen snellere versies van HyperTalk- commando's of -functies geschreven worden. Het is dus mogelijk om alle bewerkingen op een database in HyperCard uit te laten voeren door snelle, zelf geschreven commando's. Een andere belangrijke mogelijkheid van XCMD's en XFCN's is dat ze de gebruiker de kans bieden om toegang te krijgen tot non-tekst files en om HyperCard te laten communiceren met randapparatuur zoals CD-ROM, seriële poorten, videodiscs, en zelfs het AppleTalk-netwerk.

De functionaliteit van HyperCard kan naast het zelf schrijven van de externe commando's en functies ook vergroot worden door een add-on stack zoals bijvoorbeeld Reports. Deze add-on stacks bevatten veel externe commando's en functies die extra mogelijkheden bieden. Bij printen is het ook mogelijk om POSTSCRIPT commando's naar de LaserWriter te sturen. Ook dit vergroot de rapport-mogelijkheid.

Hoewel HyperCard zelf een bekrompen systeem is omdat het alleen op de Mac draait en een eigen specifieke programmeertaal heeft die alleen op objecten van HyperCard werkt, kan het door het toevoegen van XCMD's en XFCN's uitgebreid worden tot een groot en complex geheel met vele mogelijkheden.

Hoofdstuk 9.
Ideeën.

Zoals in vorige hoofdstukken al is opgemerkt, is HyperCard meer dan een database. Naast het gebruik van HyperCard bij het ontwerpen van intelligente database- systemen heb ik nog een aantal ideeën over hoe HyperCard binnen het ITK gebruikt kan worden. Tevens geven deze voorbeelden een idee van wat er allemaal nog meer kan met HyperCard.

1) Het realiseren van een hypertext-systeem.

De bedoeling is een systeem te maken waarin iemand een (samenvatting van) een tekst invoert en een lijstje bijvoegt van trefwoorden die in deze tekst gebruikt worden en die ook in andere (opgeslagen) teksten voorkomen. Ook referenties naar werk van anderen kunnen hierin opgenomen worden (voetnoten, bibliografieën). Met HyperCard is het mogelijk om deze links fysiek te leggen. Dat wil zeggen dat als iemand een volgende keer de tekst (samenvatting) leest en meer wil weten over een bepaald onderwerp, hij gewoon op het betreffende woord klikt en direct meegenomen wordt naar een bijbehorende tekst (van wie dan ook en als er een is).

Naast het leggen van de links bouwt HyperCard ook automatisch een thesaurus van de trefwoorden en bijbehorende documenten op, zodat ook hierdoor toegang tot documenten verkregen kan worden.

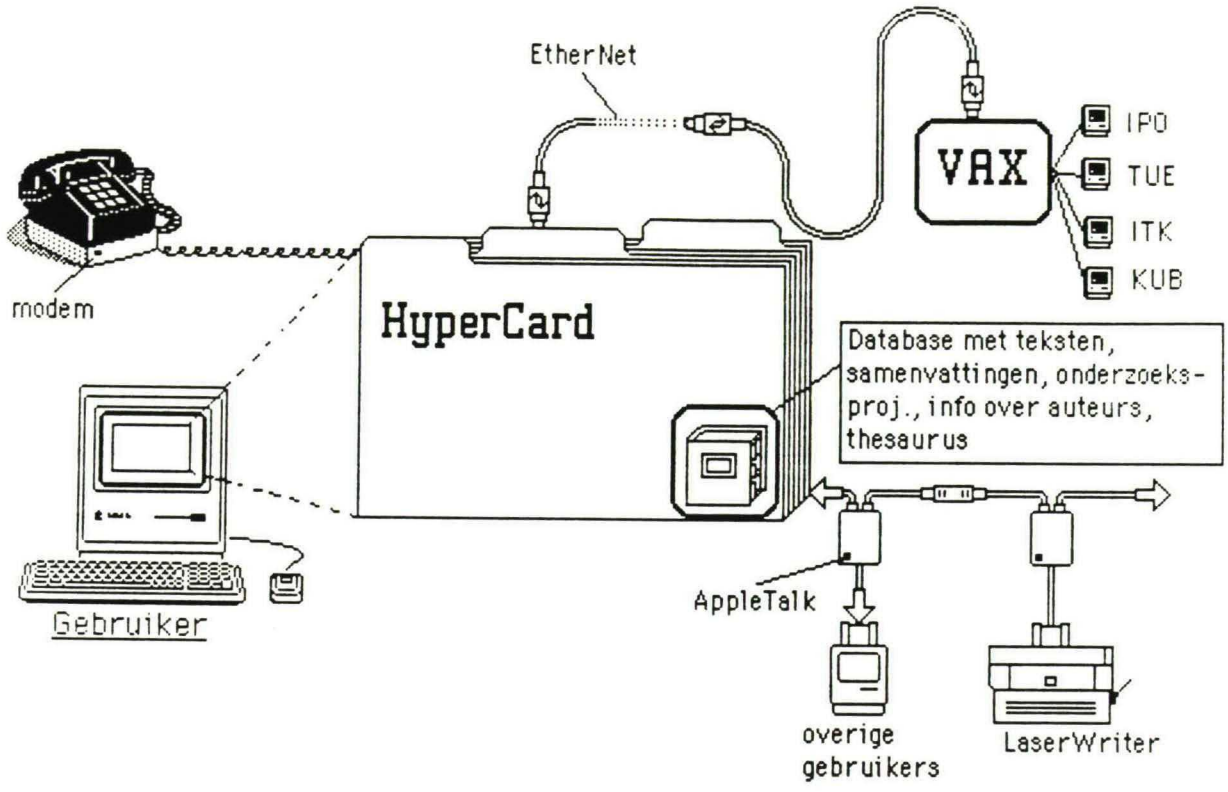
Een eerste uitbreiding hiervan is natuurlijk dat de gebruiker alleen de trefwoorden op hoeft te geven en dat HyperCard via de thesaurus zelf de links legt (tenzij de auteur slechts specifieke links wil leggen).

2) Hypermedia.

Het tweede idee bouwt eigenlijk voort op het eerste. Ook hier wordt een hypertext systeem als boven verondersteld. Nu echter bevat de 'database' van teksten ook informatie over alle onderzoeksprojecten die binnen het ITK plaatsvinden, wie er aan werkt en hoe het in verband staat met andere onderzoeken zowel binnen als buiten het ITK. Een gebruiker zou nu dus een tekst kunnen lezen en via trefwoorden naar andere teksten over dit onderwerp kunnen gaan. Ook kan hij als de tekst een samenvatting of beschrijving van een onderzoeksproject is, informatie opvragen over de auteur of uitvoerder en hoe het in verband staat met ander onderzoek. Het is zelfs mogelijk om via de telefoon of over een netwerk contact te zoeken met de auteur of uitvoerder. De informatie hoeft niet alleen te bestaan uit een geschreven tekst maar kan ook beeld (plaatjes, dia's of video) of geluid (gedigitaliseerd stemgeluid, muziek) zijn.

De andere kant op kan ook. Een auteur van een artikel zoekt over een netwerk contact met de HyperCard database en vraagt informatie over aanverwante onderzoeken en zoekt ook via HyperCard contact met een andere auteur.

In een tekening ziet het er als volgt uit:



figuur 16. Hypermedia gerealiseerd.

Het contact zoeken over het netwerk is mogelijk door een simpele druk op de knop. Het is zeker geen toekomst- muziek, want alle hardware is verkrijgbaar en HyperCard beschikt met Hypertalk en via de XCMD's over de besturings- software.

3) Time and Business Manager.

Ik zelf heb mijn 'zakenagenda' en lijst met dingen die ik nog moet doen in HyperCard zitten. Als HyperCard draait dan waarschuwt het mij op tijd dat ik een afspraak heb, eventueel kan HyperCard ook het nummer van de persoon met wie ik een afspraak heb draaien zodat ik per telefoon nog contact met hem kan zoeken over het hoe en wat. Zo ook kan HyperCard mij waarschuwen dat ik nog iets moet doen vóór een bepaalde tijd of datum. (Dit idee is nog niet geheel geïmplementeerd, maar ik ben er van overtuigd dat het kan. Tussendoor knutsel ik hier een beetje aan. Ik ben al wel zover dat HyperCard de volgende afspraak kan vinden en de agenda met de afspraak toont.)

4) Presentaties.

Dit is het meest bewijsbare en praktische idee voor het ogenblik. Zoals al eerder opgemerkt is, beschikt het ITK sinds kort over een Mac Plus, bedoeld om mooie presentaties op te maken, hetgeen nu nog met sheets gebeurt. Wat is makkelijker dan de sheets als een stapel kaarten in HyperCard te zetten en via de Mac Data Display op het scherm te tonen ? Nu zal men opmerken dat men ook gewoon in een opmaakprogramma als PageMaker de sheets als bladzijden op kan slaan. Het voordeel van HyperCard echter is dat men niet de bladzijden in volgorde af hoeft te lopen of hoeft te bladeren, maar dat met vooraf (of zelfs nog ter plekke) aangelegde links de kaarten in willekeurige volgorde kunnen worden afgelopen. Ook is een presentatie mogelijk waarvan afhankelijk van de reactie van het publiek kaarten kunnen worden getoond. Een andere mogelijkheid is om eerst een aantal kaarten die het overzicht schetsen te geven waarbij men per kaart of punt op de kaart naar kaarten kan die meer detail geven. Van al deze mogelijkheden zal ik graag een demonstratie geven als ik het praatje over HyperCard houd.

5) Dit is een uitbreiding van idee 1 en 2, naar aanleiding van de artikelen in BYTE. Hieruit blijkt dat er voor vele soorten computers (ook de binnen het ITK gebruikte) hypertext produkten op de markt verkrijgbaar zijn. Er zou eens goed over nagedacht moeten worden of er inderdaad geen netwerk binnen het ITK gebouwd moet worden waarop een hypertext systeem (commercieel of zelf ontwikkeld) draait. In de lijst die in BYTE vermeld staat staan sommige van 's werelds meest vooraanstaande onderzoeksinstituten die op dit moment onderzoek naar hypertext en de mogelijkheden daarvan doen. Ook hier moet naar mijn mening overwogen worden om een steentje bij te dragen in plaats van de resultaten af te wachten en eventueel de produkten te gaan gebruiken. Niet alleen zou dit goed zijn voor de naam van het ITK maar tevens zijn er veel mogelijkheden voor hooggekwalificeerd onderzoek omdat er nog veel moeilijkheden bestreden moeten worden.

Hoofdstuk 10.
Wat is er nodig ?

Om HyperCard te kunnen gebruiken is op z'n minst een Apple Plus met 1 Mb geheugen en 2 diskdrives van 800K nodig. Op dit moment staat er een Apple Macintosh II met kleuren monitor en 40Mb harddisk in het InfoLab. Deze Mac is eigendom van de FEW. Hij wordt gebruikt voor het schrijven van documenten en verslagen en het maken van de bijbehorende tekeningen. Verder wordt er met PROLOG op gewerkt. Op deze machine draait op dit moment zowel versie 1.0.1 (oftewel de allereerste) als de nieuwste versie 1.2 (in het Nederlands). Tot mijn grote vreugde beschikt het ITK sinds kort ook over een eigen Mac Plus (met daarbij nog een Mac Data Display). Mij is verteld dat deze machine echter voornamelijk bedoeld is om mooie verslagen en presentaties mee te maken en te geven. Hoe het maken van presentaties nog mooier kan met HyperCard heb ik beschreven in hoofdstuk 9: 'Ideeën'.

Als de ideeën zoals in hoofdstuk 9 beschreven daadwerkelijk uitgevoerd gaan worden, dan is ook een modem onmisbaar. Niet alleen om connecties te kunnen maken met andere computers binnen de afdeling of universiteit of via netwerken met andere instituten, maar ook om via bulletin boards op de hoogte te blijven wat er met HyperCard gebeurt (nieuwe releases) en wat er allemaal voor HyperCard beschikbaar is (de meeste stacks zijn freeware of shareware die via de bulletin boards verspreid worden). Als daarnaast een toepassing met behulp van een CD-ROM gerealiseerd moet worden, zal ook deze aangeschaft moeten worden.

Hoofdstuk 11.
Prestatie en Beperkingen.

In dit hoofdstuk zullen nog enkele opmerkingen over prestatie en beperkingen van HyperCard gemaakt worden. De paragraaf 'Prestatie' behandelt alleen de zoeksnelheid omdat door de hele tekst heen al de kwaliteiten en mogelijkheden van HyperCard genoemd zijn. De paragraaf Beperkingen is uitgebreider omdat hierin nog veel onbesproken nadelen of minpunten van HyperCard genoemd zullen worden.

11.1 Prestatie.

In de volgende paragraaf wordt iets gezegd over de snelheid van HyperCard, vooral dat die achteruit gaat als er grote scripts geïnterpreteerd moeten worden. Het bleek dat deze echter soms ook op is te voeren door externe commando's of functies te gebruiken.

HyperCard is erg afhankelijk van het Find-commando. Alle informatie wordt namelijk opgeslagen als text en het Find-commando wordt gebruikt om bepaalde (combinaties van) karakters, woorden of zinnen te vinden. Het Find-commando is zelf erg snel. Het heeft gemiddeld 1.5 seconde nodig om een woord te vinden in een stack van 800K (het zoeken op karakterstrings in plaats van woorden is langzamer). Dit is getest bij versie 1.0.1 op een Macintosh SE.

11.2 Beperkingen.

Zoals al in de inleiding opgemerkt is, moeten ook de minpunten van HyperCard genoemd worden om een eerlijk overzicht te kunnen krijgen. Deze paragraaf behandelt de nog onvermelde nadelen.

Het grote nadeel van HyperTalk is dat het geïnterpreteerd wordt. Er is geen compiler beschikbaar. Hierdoor kan het bij grote scripts erg langzaam worden.

Zoals ook in hoofdstuk 3 'HyperCard en database systemen' is opgemerkt, schiet HyperCard tekort in het selecteren van subgroepen van een masterlijst; rapporten maken is dus beperkt. Het is echter gebruikelijk om dit te laten doen door add-on programma's. Dit is ook zeker mogelijk in HyperCard, bijvoorbeeld door het programma Reports van Activision. Hoewel HyperCard een aantal manieren kent om snel een hardcopy van de informatie te krijgen kan een gebruiker de behoefte hebben aan meer specifieke en (voor HyperCard) moeilijke rapporten. Reports maakt dit mogelijk. Het staat een volledige specificatie van locatie en formaat van ieder veld afzonderlijk toe en tevens is het mogelijk om plaatjes en toegevoegde tekst te integreren.

Naast de tekortkomingen op het gebied van zoeken (meervoudige criteria) en het genereren van rapporten zoals hierboven genoemd, is HyperCard niet het ultieme data-manipulatie-programma. Dit komt mede doordat HyperCard slechts kan zoeken naar een enkel woord of een karakterstring. Het negeert hoofdletters, het kan niet zoeken naar strings met wild-card karakters erin en het kan ook niet zoeken naar een combinatie van items. HyperCard heeft pas in de laatste versie een zoek-en-vervang operatie toegevoegd gekregen, eerdere versies moeten het zonder stellen.

Alles wordt in Hypercard opgeslagen als tekst. Getallen worden opgeslagen als karakters en snelle converteer algoritmen zorgen ervoor dat er toch met getallen gewerkt kan worden.

Een ander nadeel, dat echter in eerste instantie slechts de ontwerper aangaat, is dat HyperCard modes kent. Als men zich bijvoorbeeld in button-mode bevindt kan men geen velden verplaatsen en ook geen enkele tekst intypen. Nog vervelender is dat er een expliciete handeling voor nodig is om over te schakelen van voorgrond naar achtergrond mode, om er voor te zorgen dat nieuwe buttons en velden tot de achtergrond behoren en niet tot de meest recente kaart. Ook het editten van een script is een van de modes van HyperCard. Dit betekent dat HyperCard helemaal stil ligt en er alleen nog maar edit commando's mogelijk zijn totdat er gestopt wordt met editten. Het is dus niet mogelijk om een id van een andere kaart na te kijken of nog even te kijken hoe het script van die andere button er ook al weer uitzag. Een gemis in HyperCard is een goede script-editor en een debugprogramma voor HyperTalk-scripts. Er zijn echter ook hiervoor add-on producten die dit deels oplossen.

Andere minpunten zijn dat er niet meerdere windows (meerdere kaarten) tegelijk bekeken kunnen worden, dat de kaarten altijd dezelfde grootte hebben, dat er niet in kleur gewerkt kan worden en het "cutting" en "pasting" van buttons en velden een vervelend werkje kan worden omdat men er maar een tegelijk kan "behandelen". Door Bill Atkinson wordt echter aan al deze punten gewerkt en zij moeten in volgende releases zijn opgelost.

Hoofdstuk 12.
Conclusie.

Dit hoofdstuk bevat mijn eigen conclusie over HyperCard en of het te gebruiken is voor de in de inleiding gestelde vraag. Tevens zijn er nog conclusies opgenomen van anderen die ik in de literatuur zoals vermeld in Appendix A tegenkwam.

Ted Nelson spreekt naast hypertext ook over wezenlijkheid (virtuality): het vermogen van een computerrepresentatie om zo transparant en toch dwingend aanwezig te zijn dat de gebruiker het bewustzijn van de computer als tussenstation verliest en zich verbeeldt met het eigenlijke ding dat gesymboliseerd is te werken.

De Macintosh excelleert in deze wezenlijkheid. Het heeft meer dan andere computers visuele, auditief en kinetische dimensies die dit ondersteunen en HyperCard maakt hier ten volle gebruik van. Het duurt niet lang voor men de muispointer beschouwt als een verlenging van de arm en hand. De wezenlijkheid van on-screen plaatjes, tekeningen en buttons als handelbare objecten vestigt zich zonder het te merken. HyperCard vergroot Macintosh's vertrouwen op metaforen (desktop, folders, verfkwasten, etc.) met zijn eigen arsenaal aan metaforen ingebouwd in kaarten en stacks. HyperCard is, in tegenstelling tot de media waar we mee zijn opgegroeid, een nog grotendeels onbekend gereedschap. Maar zijn waarde als medium wordt duidelijk als men ermee leert werken. Op een gegeven moment zal men -misschien door eigen pogingen- een begrip krijgen van hoeveel woorden of plaatjes een goede stack of button waard kan zijn.

Als deel van de conclusie of HyperCard als database te gebruiken is, wil ik nog graag verwijzen naar de woorden die Danny Goodman over het gebruik van HyperCard in [3] heeft geschreven en die ik in hoofdstuk 3 heb vermeld.

Hoewel Danny Goodman in [2] in een vergelijking tussen HyperCard en Databases de goede kanten van HyperCard belicht komt hij daar later in [3] wat op terug. In [3] zegt hij onder andere: "*It is easy to get caught up in the hullabaloo about HyperCard to the exclusion of other development systems available. HyperCard is not the do-all, end-all development system, despite its built-in powers. You still have the choice of developing in a flat file manager type of database program, a high-end relational database environment or in a traditional programming language. There are cases in which these other environments are better suited to a task than HyperCard is. Knowing when to use a database program instead of HyperCard -and vice versa- is crucial to developing an application that lives up to your expectations*".

Hieruit blijkt dat HyperCard niet een programma is om een vliegtuigreserverings-systeem mee te ontwerpen. Studies hebben echter aangetoond dat mensen vaak krachtige database programma's aanschaffen voor redelijk simpele taken die lang niet de mogelijkheden gebruiken van die programma's. Voor taken zoals deze en voor prototyping van ingewikkelde databases is HyperCard een aantrekkelijk alternatief.

Mijn eigen conclusie met betrekking tot het onderwerp van deze tekst is dat hoewel er enige nadelen aan HyperCard kleven wat betreft het implementeren van relationele database-systemen, deze met niet al te veel moeite en met niet al te veel prestatievermogenverlies grotendeels zijn op te lossen. Ik denk dat HyperCard zeker mogelijkheden biedt, maar het zal zo simpel mogelijk gehouden moeten worden, want als de scripts groter worden, hetgeen vaak gebeurt bij ingewikkelde programma's, holt het prestatievermogen achteruit. Dit wat betreft het maken van relationele databases.

Ik denk tevens dat het de moeite waard is om te proberen om andere soorten databases te ontwerpen, die wat HyperCard betreft meer naar de object-georiënteerde database-kant gaan. In verband hiermee verwijs ik ook graag naar het eind van hoofdstuk 5 waar ik al een conclusie heb getrokken.
Tevens biedt dit de mogelijkheid om zinvol op zoek te gaan naar de manier waarop HyperCard als intelligente database gebruikt kan worden. Ik ben ervan overtuigd dat er mogelijkheden liggen en als voorbeeld wil ik nog eens wijzen op het gebruik van HyperCard in QFE.

Meer zekerheid bestaat er over het nut van het toepassen van HyperCard met/bij bestaande databases. HyperCard is het meest gebruikersvriendelijke programma dat ik ken en als front-end of user-interface kan het alleen maar voordeel opleveren.

Het probleem met HyperCard en de Macintosh is natuurlijk dat het geheugen en de opslagmogelijkheden te beperkt zijn om een grote database te ontwerpen (tenzij het een read-only database wordt, dan kan men met CD-ROMs van 500 Mb een eind komen). Men komt dan ook al snel uit op een systeem zoals het beschreven QFE waar het database schema (gebaseerd op een semantisch model) en de user-interface in HyperCard ontworpen kunnen worden en de laatste ook met HyperCard als hypertext-systeem werkt.

En tot slot, als databases en/of user-interfaces dan niet de moeite waard zijn, denk ik dat er nog andere interessante mogelijkheden voor HyperCard binnen het ITK liggen, zoals de ideeen die ik in hoofdstuk 9 gegeven heb.

Ik zou graag afsluiten met een naar mijn mening mooie conclusie over wat HyperCard kan en is, alhoewel niet direct van toepassing op het onderwerp, die wordt geformuleerd door Greg Williams in een artikel in Byte [4]: "*Years ago, new automated dialing circuitry in the telephone system allowed the expansion of phone service past the limit imposed by the finite number of human operators. In effect, you became your own operator. HyperCard has a similar potential: to allow the creation of a far larger number of useful programs than is possible from the work of a finite number of professional software developers. We will become our own programmers, and we will be able to create new programs and modify existing ones to meet our needs*".

APPENDIX A. LITERATUURLIJST

[1] Using HyperCard, From Home to HyperTalk
door: Tay Vaughan, Que Corp., 1988.

[2] The Complete HyperCard Handbook 2nd ed.,
door: Danny Goodman, Bantam Books, 1988.

[3] Danny Goodman's HyperCard Developer's Guide.
door: Danny Goodman, Bantam Books, 1988.

[4] HyperCard,
door: Greg Williams,
in: BYTE, December 1987, blz. 109-117.

[5] BYTE, Oktober 1988:
* A grand vision
door: J. Fiderio, blz. 237-244.
* From text to Hypertext
door: M. Frisse, blz. 247-253.
* The right tool for the job
door: M. Begeman and J. Conklin, blz. 255-265.

[6] Foundations of semantic query optimization for deductive databases,
door: U.S. Chakravarthy, J. Grant and J. Minker,
in: Foundations of deductive databases and logic programming,
door: J.Minker (ed.), Morgan Kaufmann Publ. Inc., 1988, blz. 243-273.

[7] Towards Models for Practical Reasoning about Conceptual Database Design,
door: prof. R.A. Meersman,
in: Data and Knowledge
door: R.Meersman and A.Sernadas (ed.), Norht Holland Publ. Co., 1988.

[8] Hypertext: An introduction and survey,
door: J. Conklin,
in: IEEE Computer, September 1987, blz. 17-41.

[9] CACM (Communications of the ACM) vol. 31, nr. 7, Juli 1988:
* Hypertext
door: J.B.Smith and S.F.Weiss, blz. 816-819.
* KMS: A distributed hypermedia system for managing knowledge in organizations,
door: R.M.Akscyn, D.L.McCracken, E.A.Yoder, blz. 820-835.
* Reflections on NoteCards: seven issues for the next generation of hypermedia systems,
door: F.G.Halasz, blz. 836-852.
* HAM: A general purpose Hypertext Abstraction Machine,
door: B.Campbell and J.M.Goodman, blz. 856-861.

* Abstraction mechanisms in hypertext,
door: P.K.Garg, blz. 862-870.

[10] QFE: A query interface using a hypertext approach based on semantic contexts,
door: T. Estier, G. Falquet

[11] DSDS: An outline and prototypical implementation of a deductive semantic database system,
door: E.M. Verharen,
afstudeerverslag TU Twente, mei 1988.

[12] Microcomputers for Information management, vol. 5, nr.1, Maart 1988:
* Design considerations for hypermedia systems
door: B.R.Gaines and J.N.Vickers.
* A comparison of Books and hypermedia for knowledge-based sports coaching,
door: J.N. Vickers and B.R. Gaines.
* Search-Aid: A mark-up language and interpreter to provide on-line documentation for database searchers,
door: W. Leigh, N. Poz and P. Wen-Sheng Chiang.
* MicroWatch
door: Ching-chuh Chen.

[13] Free text databases on small computers,
door: H. Paymans,
in: RAIO-88 Conference on User-oriented content-based text and image handling, MIT, 1988, bz. 491-499.

[14] Hypertext: wat is het en wat is het (nog) niet,
door: C.A. Benschop,
in: NVKI-Nieuwsbrief, vol. 6, nr. 1, Februari 1989, blz 5-8.

[15] Getting it out of our system,
door: T.H. Nelson,
in: Information Retrieval: A critical review,
door: G. Schechter (ed.), Thompson Books, Wash.D.C., 1967.

[16] A conceptual framework for the augmention of man's intellect,
door: D.C. Engelbart,
in: Vistas in Information Handling, vol 1, Spartan Books, London, 1963.

APPENDIX B. TEKENINGEN-OVERZICHT

ITK: P.O. BOX 90153 5000 LE TILBURG THE NETHERLANDS