Software Design Document

Vilas Pultoo, Judith van Stegeren, Patrick Uiterwijk
Niels van der Weide

29 april 2011

1 Introductie

Dit ontwerpdocument heeft als doel om de architectuur en het ontwerp van de applicatie 'CampNav' te beschrijven.
De aard van het project is een programma dat routes in de campus berekent. De omvang van deze routeplanner is het Huygensgebouw. De beperking van het programma is het gebrek voor ondersteuning op andere locaties van de campus.

2 Overzicht van het systeem

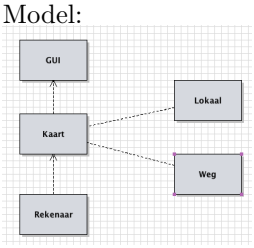
Het product heeft als functie om mensen te helpen om lokalen te vinden binnen het Huygensgebouw.
De context van het project is de internationale studenten die maar voor korte tijd bij ons op de campus verblijven. Zij hebben moeite om lokalen te vinden. Voor sommige van deze mensen is de drempel om hulp te vragen veel hoger door een taalbarrière.
We benaderen het project vanuit de studenten.

3 Architectuur van het systeem

3.1 Architectuur

We gebruiken het object-georiënteerd paradigma. De verschillende modules zijn objecten.
Er zijn vijf belangrijke onderdelen: de GUI (wat de gebruiker te zien krijgt), het algoritme (deze berekent het kortste pad) en de kaart. De kaart bestaat uit knopen (lokalen) en paden (wegen tussen lokalen). De GUI toont de kaart en in de GUI vraagt invoer van de gebruiker. Zodra de gebruiker de invoer heeft gegeven, dan wordt het algoritme aangeroepen. Dit algoritme berekent het kortste pad en geeft het pad door aan de kaart, zodat de GUI het pad kan tekenen. De knopen hebben als verantwoordelijkheid om informatie van de lokalen bij te houden en wegen om de informatie van de paden tussen wegen bij te houden.
Samenvatting:
1. GUI. Verantwoordelijkheden: pad tonen, invoer vragen.
2. Algoritme. Verantwoordelijkheden: kortste pad berekenen.
3. Kaart. Verantwoordelijkheden: informatie van de kaart bijhouden.

4. Knoop. Verantwoordelijkheden: informatie van lokalen bijhouden.
5. Wegen. Verantwoordelijkheden: informatie van paden tussen lokalen bijhouden.
Model:



3.2 Redenen voor ontwerp

Is dit ontwerp wel elegant?
Er is een goed onderscheid van klassen. Elke klasse heeft 1 of 2 'abstracte' verantwoordelijkheden. Dat is goed voor het ontwerp.
Het lijkt raar om de rekenaar een aparte klasse te maken. Je kunt dit namelijk ook bij de kaart toevoegen. Het nadeel hiervan is dat je daardoor het werkt van de rekenaar afhankelijk maakt van de representatie van de kaart. Als je dus de datastructuur van de kaart veranderd, dan moet de rekenaar veranderen. Door de rekenaar een aparte klasse te maken, werkt het onafhankelijk van de representatie van de kaart.
Verder kun je vragen stellen over de representatie van een kaart. We hebben ervoor gekozen om apart knopen en wegen te representeren. Omdat een kaart een wiskundige graag is en elke graaf te representeren is met een matrix, konden we het ook representeren met een matrix. Dat is echter onhandig, want je moet ook dingen weten over de knopen. Daardoor is het handig om een knoop-object te maken.

4 Data van het programma

4.1 Beschrijving van de data

De belangrijkste data van het programma is de kaart en het pad.
Er zijn dus lokalen en wegen. De wegen moeten de lengte bijhouden, het beginlokaal en het eindlokaal. Van het lokaal moet je de naam bijhouden en de wegen. De namen van de lokalen zijn 'strings', de buren worden bijgehouden in een lijst van wegen en de lengte van de wegen zijn 'integers'. Het begin- en eindpunt van de wegen zijn lokalen.
Een kaart bestaat dus uit een aantal knopen. De knopen weten zelf wat de wegen zijn, dus hoef je de wegen niet bij te houden.
Nu is nog de vraag of de kaart 'hardcoded' in het programma moet staan of gelezen moet worden via een file. Als je met files werkt, dan is het programma dynamischer en kun je het makkelijker aanpassen. Dat is een significant voordeel.

4.2 Woordenboek van de data

1. Planner. Attributen: een kaart. Methodes: bepaal kortste route van punt x naar y (parameters: de punten).
2. Kaart. Attributen: lijst van knopen. Methodes: geef kaart (geen parameters) en geef neighbours van een knoop (parameter: de knoop).
3. Knoop. Attributen: naam, lijst van wegen. Methodes: geef naam (geen parameters), geef buren (geen parameters) en voeg een buur toe (parameter: een knoop).
4. Weg. Attributen: beginpunt, eindpunt, afstand. Methode: geef afstand, geef eindpunt, geef beginpunt. De methoden hebben geen parameters.

5 Ontwerp van componenten

Alleen de rekenaar is een component wat iets doet. De rest van de component hebben een administratieve rol.
De rekenaar maakt gebruik van het algoritme van Dijkstra. Dijkstra hield van simpelheid, dus dit moet dus ook simpel worden. Voor de simpelheid moet je ervoor zorgen dat de rekenaar weinig objecten kent. Daardoor hoeft het weinig te communiceren. De rekenaar moet praten met de knopen, want de rekenaar wil de buren van de knopen weten. Tevens moet de rekenaar praten met de kaart, omdat de kaart de route wil weten.

6 Human Interface Design

6.1 Overview of User Interface

De gebruiker kan *on startup* kiezen naar welke plaats hij wil navigeren. Dat kan door iets in te typen of een lijst met alle mogelijkheden op te vragen. Vervolgens bepaalt de locator waar hij staat. Vanaf dat moment wordt dat weergegeven on screen, met een kaart van boven, á la googlemaps. Er wordt een route uitgestippeld van de plaats waar hij staat naar waar de gebruiker wil zijn. De route wordt over de kaart heen geprojecteerd, als een gekleurde lijn. Er zijn twee knoppen altijd in beeld: quit en new destination. *quit* zorgt ervoor dat de applicatie wordt beeindigd, en *new destination* stelt de gebruiker in staat om een nieuwe plek van bestemming te kiezen.

6.2 Screen images

6.3 Screen images and actions

We hebben geprobeerd het interface zo duidelijk mogelijk te houden. Grote knoppen zijn een must voor goede android-apps, dus we hebben een maximum van twee knoppen met een duidelijk lettertype. Bij quit en new destination hebben we ook nog de kleuren om het te verduidelijken.

Image	Action
Quit	De applicatie wordt beeindigd
New destination	De gebruiker krijgt de mogelijkheid om een bestemming in te voeren
Huygensgebouw	De gebruiker krijgt een lijst met lokalen van het huygensgebouw om uit te kiezen
Show all locations	De gebruiker krijgt een lijst met alle lokalen die in de database staan (in ons geval: ook alleen de lokalen van het Huygensgebouw
Cancel	De gebruiker kan nogmaal de keuze maken tussen een bepaald gebouw en een lijst van alle lokaties