Stichting NIOC en de NIOC kennisbank

Stichting NIOC (www.nioc.nl) stelt zich conform zijn statuten tot doel: het realiseren van congressen over informatica onderwijs en voorts al hetgeen met een en ander rechtstreeks of zijdelings verband houdt of daartoe bevorderlijk kan zijn, alles in de ruimste zin des woords.

De stichting NIOC neemt de archivering van de resultaten van de congressen voor zijn rekening. De website www.nioc.nl ontsluit onder "Eerdere congressen" de gearchiveerde websites van eerdere congressen. De vele afzonderlijke congresbijdragen zijn opgenomen in een kennisbank die via dezelfde website onder "NIOC kennisbank" ontsloten wordt.

Op dit moment bevat de NIOC kennisbank alle bijdragen, incl. die van het laatste congres (NIOC2023, gehouden op donderdag 30 maart 2023 jl. en georganiseerd door NHL Stenden Hogeschool). Bij elkaar bijna 1500 bijdragen!

We roepen je op, na het lezen van het document dat door jou is gedownload, de auteur(s) feedback te geven. Dit kan door je te registreren als gebruiker van de NIOC kennisbank. Na registratie krijg je bericht hoe in te loggen op de NIOC kennisbank.

Het eerstvolgende NIOC vindt plaats op donderdag 27 maart 2025 in Zwolle en wordt dan georganiseerd door Hogeschool Windesheim. Kijk op www.nioc2025.nl voor meer informatie.

Wil je op de hoogte blijven van de ontwikkeling rond Stichting NIOC en de NIOC kennisbank, schrijf je dan in op de nieuwsbrief via

www.nioc.nl/nioc-kennisbank/aanmelden nieuwsbrief

Reacties over de NIOC kennisbank en de inhoud daarvan kun je richten aan de beheerder:

R. Smedinga kennisbank@nioc.nl.

Vermeld bij reacties jouw naam en telefoonnummer voor nader contact.

ARTIKEL

Apps in curricula

Apps en web-apps als onderwerp in het (universitaire) informaticacurriculum

Door: ir. Sylvia Stuurman, Open Universiteit.
Met medewerking van: ir. Harry Passier; Open Universiteit.
Trefwoorden: apps, web-apps, programmeeronderwijs.

Voor studenten is het aantrekkelijk om apps te leren bouwen: het is leuk om iets te maken dat je op je eigen mobiele telefoon kunt draaien, en het opent de mogelijkheid om te proberen apps te bouwen voor een groot publiek. Een voorbeeld of een opdracht waarin met apps wordt gewerkt binnen een vak, werkt dus stimulerend voor studenten. Binnen de afdeling Softwaretechnologie van de faculteit Informatica van de Open Universiteit hebben we daarom onderzocht of en hoe we apps en web-apps zouden kunnen gebruiken in ons curriculum. Niet als ondersteuning, maar als onderwerp voor studenten. Zijn er belemmeringen? Wat zijn de mogelijkheden?

1. Wat zijn apps?

De term apps is in feite een afkorting van applicaties, maar heeft in de praktijk een beperktere betekenis gekregen: wat met de term apps wordt aangeduid zijn applicaties die specifiek geschreven zijn voor mobiele apparaten als smartphones en tablets. Kenmerkend voor mobiele apparaten ten opzichte van niet-mobiele apparatuur is onder andere dat mobiele apparaten een internetverbinding kunnen hebben die elk moment kan uitvallen. Andere kenmerken zijn zaken als input via een touch screen, de mogelijkheid van locatiebepaling, interactie met belfuncties, een klein scherm met soms juist weer een enorme resolutie, enzovoort.
Zoals de term apps voor een bijzonder type applicaties staat, staat de term web-apps voor webapplicaties die specifiek gericht zijn op mobiele apparaten. De term apps wordt ook wel als verzamelnaam gebruikt. In dat geval staat de term native apps voor applicaties die direct op het operating system van de smartphone of tablet draaien, terwijl web-apps gebruik maken van een browser op een mobiel apparaat.

2. Native apps, mogelijkheden in het curriculum

Web-apps zijn eenvoudiger toe te passen: er is weinig extra studietijd nodig om ermee te kunnen ontwikkelen en de apps lenen zich uitstekend als illustratie van geavanceerde technieken in JavaScript. Ze zijn daarnaast erg bruikbaar om studenten te leren netjes te programmeren.
Voor native apps voor het Android-platform is een ontwikkelomgeving beschikbaar voor Java. Voor een curriculum als het onze, waarin Java de belangrijkste programmeertaal is, is dat aantrekkelijk. We hebben daarom de mogelijkheid onderzocht of we het bouwen van een native app zouden kunnen gebruiken bij – bijvoorbeeld – een integrerend practicum.

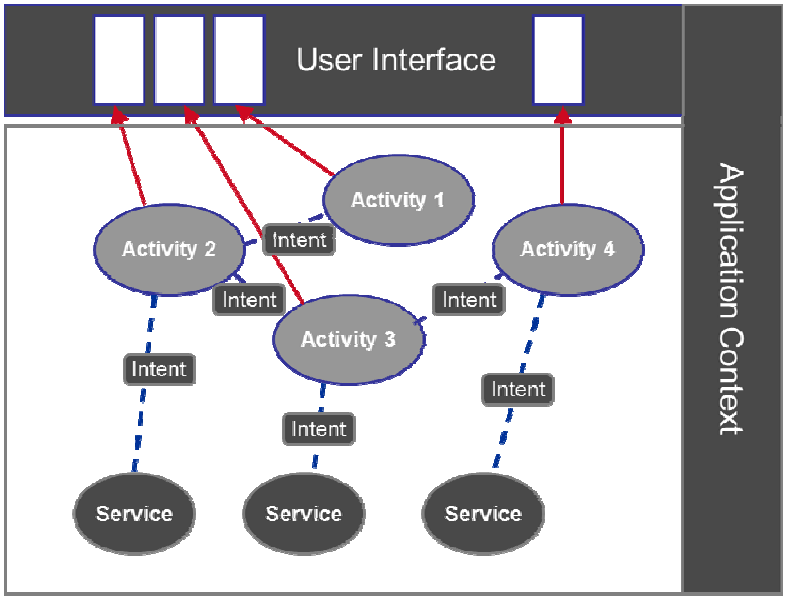
2.1. Tooling

Wat de tooling betreft zijn er niet heel veel problemen. De Android Software Development Kit is inclusief een plug-in voor Eclipse gratis beschikbaar en er is een Android-emulator beschikbaar om de

app mee te testen. Het nadeel is wel dat de versies van zowel Android als de SDK elkaar snel opvolgen en dat er vaak sprake is van incompatibiliteiten. Het is daardoor lastig tot onmogelijk om uitgebreide handleidingen ter beschikking te stellen. Aanwijzingen zullen gegeven moeten worden in de vorm van links naar documentatie en die links moeten zeer regelmatig worden nagelopen. De documentatie is sinds de start van ons onderzoek overigens enorm verbeterd. Op dit moment is de documentatie beslist geschikt om studenten mee op weg te helpen.
Een ander lastig aspect van de tooling is dat de emulator erg traag is (vooral op wat oudere systemen), wat de ontwikkeltijd flink verlengt.

2.2. Concepten

Android-apps kennen enkele concepten die niet in Java-applicaties voorkomen. Voordat een student een native app kan ontwikkelen, zullen die concepten eerst moeten worden bestudeerd. Een Android-app is bijvoorbeeld opgedeeld in een aantal activities, die in een stack boven elkaar liggen, zodat de onderliggende activity actief wordt wanneer de gebruiker of het systeem de bovenste afsluit of tijdelijk onderbreekt.
Android-apps communiceren via intents: ook een concept dat studenten niet kennen uit de gevolgde programmeercursussen. De communicatie verloopt asynchroon en kan anoniem zijn, analoog aan een eventbus. Dit vereist een andere manier van programmeren dan studenten gewend zijn. Dat geldt ook voor het feit dat Android-apps sterk event-gestuurd werken. De architectuur van een Android-app, weergegeven in figuur 1, is dus anders dan wat studenten kennen. Door de nieuwe concepten is ook de API een stuk ingewikkelder dan studenten van 'gewoon' Java gewend zijn.
Threads spelen een belangrijke rol in apps. Bijvoorbeeld het ophalen van data van een server wordt uitgevoerd in een aparte draad, zodat de gebruiker ondertussen door kan gaan met het gebruiken van de app. Het aantal draden binnen een applicatie neemt al snel toe, waartussen veelal ook synchronisatie moet plaatsvinden.



Figuur 1. De architectuur van een Android-app.

Een probleem van deze concepten is ook dat niet duidelijk is hoe ze in UML te representeren: het is daardoor lastig om apps te modelleren in UML. Wanneer het opstellen van een ontwerp tot de opdracht behoort, zal de docent zelf eerst moeten uitzoeken hoe dat is aan te pakken.

Wanneer het bouwen van een app dus binnen de context van een integrerend practicum plaatsvindt, moet er rekening mee worden gehouden dat studenten een deel van de beschikbare tijd zullen moeten besteden aan het leren begrijpen van deze concepten, de architectuur van een Android-app, en aan het leren werken met de API.

3. Web-apps, mogelijkheden in het curriculum

Web-apps zijn in principe te bouwen met kennis van HTML, CSS en JavaScript die studenten hebben opgedaan in een basiscursus webapplicaties. Er zijn conventies voor web-apps, die rekening houden met het feit dat de vingers in plaats van de muis worden gebruikt en dat de applicaties snel moeten reageren. Verreweg de snelste manier om studenten te leren werken met die conventies is ze een framework aan te bieden als JQuery Mobile. In vergelijking tot de extra studie die nodig is voor het ontwikkelen van een Android-app kost het erg weinig tijd om te leren werken met zo'n framework. Het kenmerk dat de verbinding gemakkelijk weg kan vallen, houdt in dat er in web-apps vaker gewerkt zal worden met een client-side-database dan in andere webapplicaties. Ook zullen er eerder web workers (threads voor webapplicaties) gebruikt worden dan in andere webapplicaties. Web-apps lenen zich dus om die geavanceerde technieken te oefenen.
Doordat web-apps over het algemeen vrij uitgebreide webapplicaties zijn, lenen ze zich goed om studenten te oefenen met event-driven-programmeren, en vooral ook met een modulaire opzet: een lagenstructuur. Juist bij web-apps is het daardoor van belang om netjes te programmeren in JavaScript.

Figuur 2. Firefox OS voor mobile phones.

Er zit beslist toekomst in Web-apps. Er bestaat al een Firefox Operating System(OS) voor smartphones (figuur 2) waarop alleen Web-apps gebruikt kunnen worden. Een probleem bij web-apps is, net zoals bij native apps, dat het niet voor de hand ligt hoe je UML kunt gebruiken om apps te ontwerpen.

Conclusie

Native apps vormen een aantrekkelijke techniek voor studenten tijdens een integrerend practicum: enerzijds is het een nieuwe en actuele ontwikkeling, anderzijds spelen een aantal complexe concepten een rol. Er zijn een aantal haken en ogen waar rekening mee moet worden gehouden. De

snelle opeenvolging van versies met soms grote veranderingen maken het onmogelijk om zelf leermateriaal te schrijven. De ontwikkelomgeving kan vertragend werken. Een behoorlijk deel van de tijd van zo'n practicum zal moeten worden ingeruimd voor het leren van de benodigde concepten. Toch hebben we het idee dat voor studenten die er in geïnteresseerd zijn, de voordelen opwegen tegen de nadelen.
Voor wat betreft Web-apps liggen de zaken iets minder gecompliceerd. Er is weinig extra studietijd nodig om ermee te kunnen ontwikkelen. Web-apps lenen zich uitstekend als illustratie van geavanceerde technieken in JavaScript. Web-apps lenen zich ook uitstekend om studenten te leren netjes in JavaScript te programmeren, met gebruik van modules en volgens een gelaagde architectuur. In een practicum waar ook de serverside aan bod komt, kunnen ze ook prima worden ingezet.

Discussie
In de discussie na de presentatie kwamen een aantal interessante mogelijkheden voor apps ter sprake. Een van de mogelijkheden is om Android-apps te gebruikten binnen het programmeeronderwijs door een framework aan te bieden waarbij studenten alleen zaken hoeven aan te vullen. De noodzaak om de nieuwe concepten aan te leren vervalt dan, terwijl de student wel de voldoening krijgt om een 'echte' app te ontwikkelen. De betreffende docent heeft nog geen ervaring met de consequenties van nieuwe versies van Android. Een mogelijkheid is om een open-source-project te maken van het framework, dat bijgehouden kan worden door docenten die van deze mogelijkheid gebruik willen maken.
Een andere mogelijkheid is om juist het ontdekken van de nieuwe concepten, de architectuur en de API als 'leermoment' te gebruiken door het ontwikkelen van een app als opdracht te gebruiken voor het vak Software engineering. Studenten leren op die manier aan den lijve wat het betekent om voor een opdracht een nieuwe techniek te gaan gebruiken.
Ook was er de suggestie om voor het modelleren te kijken naar modeleertechnieken voor real-time systemen: wellicht zijn sommige modelleertechnieken op dat gebied geschikt om activities en intents mee te modelleren.
Tenslotte was er de suggestie om studenten aan de hand van apps te laten nadenken over mogelijkheden in de toekomst: wat kun je voor apps ontwikkelen wanneer er nieuwe sensoren worden ontwikkeld, zoals een geursensor?

Wilt u reageren op dit artikel of de presentatie? Neem dan contact op met:
Ir. S., Stuurman; Universitair docent, Open Universiteit Nederland
Sylvia.Stuurman@ou.nl