

USPOREDBA RAZVOJA MOBILNE APLIKACIJE NA NOKIA QT I MICROSOFT WP7 PLATFORMAMA

COMPARING THE DEVELOPMENT OF MOBILE APPLICATION ON NOKIA QT AND MICROSOFT WP7 PLATFORMS

mag.inf. Zlatko Stapić, Josip Vincek, Zoran Šaško, Miroslav Zver

SAŽETAK:

Globalna popularnost nekoliko različitih platformi za razvoj aplikacija za mobilne uređaje rezultirala je činjenicom da razvojni timovi sve češće razvijaju sustave koji na istoj servisno orijentiranoj infrastrukturi poslužuju više mobilnih aplikacija razvijenih za uređaje pogonjene različitim operacijskim sustavima. Iako je u osnovi nemoguće „pomiriti“ razvoj za Nokia Qt podržane uređaje i Microsoft Windows Phone 7 podržane uređaje, ovaj rad prikazuje sličnosti i razlike spomenutih platformi na primjeru razvoja sustava za geo-lokacijski podržano kupovanje. Rad predstavlja i opisuje spomenute platforme, te ističe ključne elemente razvoja aplikacije u jednom i drugom razvojnom okruženju.

ABSTRACT:

The overall popularity of a several different platforms for the mobile phones application development has resulted in the fact that developers are increasingly developing systems with a single service oriented infrastructure serving multiple applications developed for a mobile devices powered by a various operating systems. Although it is basically impossible to "reconcile" the development for the Nokia Qt supporting devices with the development for the Microsoft Windows Phone 7 supporting devices, this paper shows similarities and differences between these platforms and is based on the example of development of geo-location supported purchase enhancement software. The paper will describe these platforms, and point out key elements of developing applications in both development environments.

1. UVOD

Projekt SmartBuy nastao je iz ideje da se kupcima olakša potraga za proizvodima koji su trenutno na akciji, a nalaze se u korisnikovoj blizini. Prije pojave Internet trgovina (*engl. web shop*) traženje jeftinog proizvoda svodilo se na hodaње od dućana do dućana i često na vožnju po cijelom gradu. Nakon što su se pojavile Internet trgovine, taj proces se olakšao jer se ponuda i cijene proizvoda mogu provjeriti preko Interneta. Ipak, još uvijek veliki broj trgovina nema vlastitu Internet stranicu, a osim toga jako je teško pratiti i biti informiran o svim akcijama. Upravo zbog toga, ideja vodilja projekta SmartBuy je omogućiti potragu za jeftinim proizvodima u blizini na temelju naše trenutne geolokacije.

Arhitektura SmartBuy sustava je prikazana na slici 1. Sustav se sastoji od web servera (na kojem se nalaze web servisi), servera sa bazom podataka MSSQL 2008 R2, web i mobilne aplikacije te od pružatelja usluga za mape. Kod izbora baze jedan od važnih kriterija je bila i podrška za spremanje i pretraživanje geolokacijskih podataka. Kako MySQL trenutno ima nerazvijenu podršku za pohranu geolokacije, odabrali smo MSSQL 2008 R2. Web aplikacija nam služi za administriranje baze podataka, te je isto tako isporučujemo zainteresiranim prodavaonicama kako bi mogle u bazu upisivati proizvode koji su trenutno na akciji. Naravno, tim prodavaonicama prethodno kreiramo korisnički račun pomoću kojeg se netko od zaposlenika može ulogirati u aplikaciju i ažurirati popis proizvoda na akciji i potrebne detalje o proizvodu i samom iznosu popusta.

U ovom radu pokazat ćemo usporedbu razvoja spomenute mobilne aplikacije na dvije bitno različite platforme. Upoznajući čitatelja sa Nokia Qt razvojnim

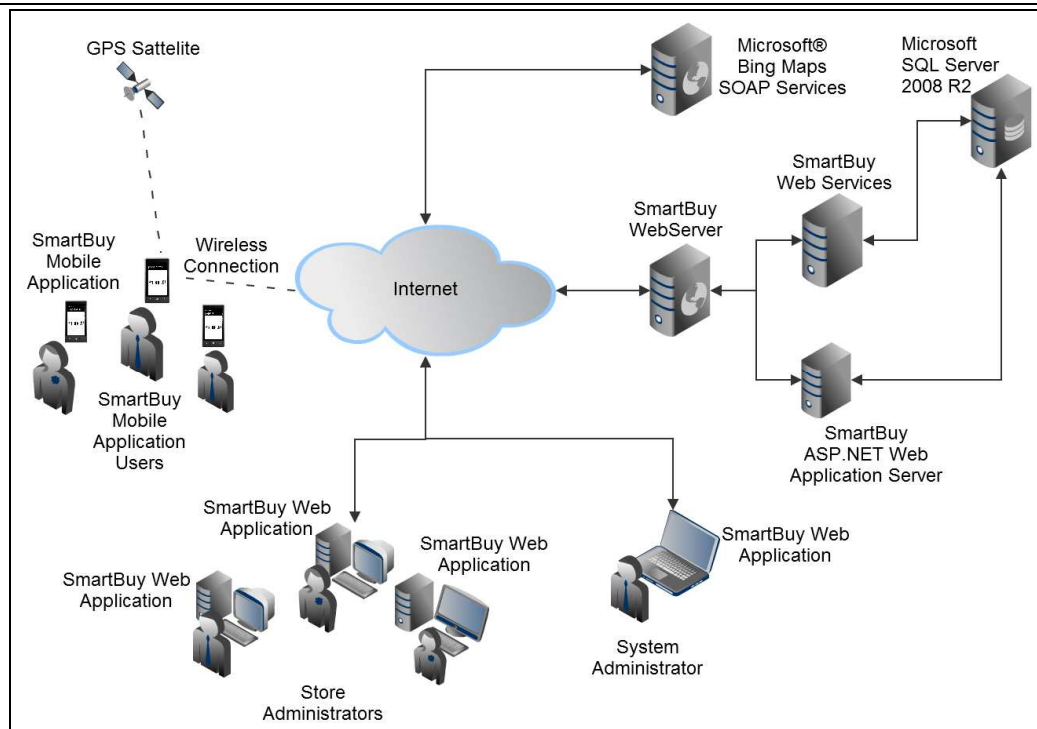
okruženjem i sa Microsoft Windows Phone 7 razvojnim okruženjem, ovaj rad ima za cilj prikazati prednosti i nedostatke spomenutih platformi, ali i u kratkim crtama prikazati sve bitne elemente razvoja aplikacije koja se oslanja na prikazanu servisno orijentiranu infrastrukturu.

2. NOKIA QT

Sve bržim razvojem tržišta pred programere se postavljaju i sve značajniji zahtjevi za razvoj programskog proizvoda koji će moći raditi na više platformi. Kako bi ispunili ovaj zahtjev, programeri moraju pronaći jedinstveni okvir (*engl. framework*) koji će sadržavati biblioteke klase za omogućavanje rada programa na različitim platformama. Qt aplikacijski okvir, koji sadržava razvojne alate i C++ biblioteke klase, pruža stabilno okruženje u kojem razvojni programeri mogu kreirati bogata, vizualno atraktivna korisnička sučelja koja se prilagođavaju najnovijim platformama i internetskim tehnologijama.

Pristupom „napiši jednom, izvršavaj bilo gdje“ Qt C++ okvir na temelju jednostavnog prevođenja programskog koda (*engl. recompilation*), stvara aplikacije koje mogu raditi na Windows, Mac OS X, Linux, Solaris, HP-UX te na mnogim verzijama Linux operacijskog sustava. Qt aplikacije se isto tako mogu izvršavati na različitim verzijama ugrađenog (*engl. embedded*) Linux-a, Symbian-a ili na Windows CE platformama.

Prva verzija Qt je izdana 1994. godine, kao skup C++ razvojnih alata i C++ biblioteka klase te je od tada evoluirao u potpuno novi oblik sa vlastitom integriranom razvojnom okolinom i SDK-om. Qt aplikacije se razvijaju u Qt Designer alatu koji zapravo predstavlja prilagodljiv alat za razvoj korisničkog sučelja. Snaga Qt okvira se



Slika 1. Arhitektura sustava

nalazi u bogatom skupu biblioteka koje pružaju razvojnim programerima mogućnost razvoja grafičkog sučelja (*engl. graphical user interface - GUI*), dohvaćanje i spremanje podataka u različite baze podataka (Oracle, Microsoft SQL Server, IBM DB2, PostgreSQL, MySQL, Borland Interbase, SQLite, Adaptive Server te ostale baze podataka prilagođene prema ODBC), komunikaciju preko mreže (pokretanje web stranica, pozivanje web servisa i sl.), multimedijske mogućnosti, razvoj 2D i 3D grafičkih elemenata, ECMAScript, web programiranje te složene XML operacije (SAX i DOM klase za učitavanje i manipuliranje podacima spremljenim u XML formatu) [7].

QT također uključuje bogati skup „kontrola“ (*engl. widgeta*) u razvojnim okruženjima za Windows, koji pružaju standardnu funkcionalnost grafičkog sučelja. U QT je uključen inovativni koncept za komunikaciju između objekata, pod nazivom „signals and slots“, odnosno signali i slotovi, kojima se zamjenjuje zastarjeli i nesiguran način povratnih poziva koji je prisutan kod nekih starijih (*engl. legacy*) okvira. QT isto tako pruža konvencionalni model za upravljanje korisničkim ili automatski pokrenutim događajem.

QT aplikacije mogu proširiti svoju funkcionalnost uključivanjem različitih plugin komponenata (npr. upravljački program za bazu podataka, stilovi, različite kontrole i sl.) ili dinamičkih biblioteka. Za razvoj aplikacija koje trebaju podržavati višejezičnost, postoji i alat QT Linguist.

Prednosti ovog razvojnog okruženja su [3]:

- *Jedinstveni programski kôd koji se izvršava na različitim platformama* – QT aplikacija i UI okvir pružaju mogućnost implementacije programskog koda sa jedne ciljane razvojne platforme (poput platforme za stolna računala), na drugu (poput ugrađenog OS-a) bez potrebe za ponovnim prilagođavanjem programskog koda za ciljanu platformu.
- *Kraće vrijeme potrebno za lansiranje proizvoda na tržište* – Prilikom razvoja aplikacije za instalaciju na različita okruženja, jedinstveni programski kôd ubrzava završavanje projekta u odnosu na ciljane platforme.

- *Manji troškovi održavanja* – Zbog jedinstvenog programskog koda za sve ciljne platforme, programer ne treba održavati sve programske proizvode za različite platforme već samo jedan programski proizvod, čiji programski kôd će se nakon odgovarajućeg prevođenja moći izvršavati u ciljanoj platformi.
- *Postizanje više sa manje* – Velik broj klasa koje razvojnim programerima olakšavaju razvoj bogatih aplikacija.

Korištenjem QT okvira za razvoj mobilnih aplikacija postižu se značajne uštede u vremenu i resursima s ciljem olakšavanja odvijanja poslovnih procesa preko mobilne aplikacije.

3. WP7 PLATFORMA

Microsoft se bavi razvojem mobilnih platformi već petnaestak godina, počevši sa uređajima baziranim na Windowsima CE, koji su svjetlo dana ugledali 1996. godine. Tijekom nekoliko godina, nastali sustavi počeli su konvergirati u Windows Mobile, vodeći se principom „dostavljanja“ osobnog računala u vaš džep. Nove funkcionalnosti koje su dodavane većinom su bile vođene potrebama samih uređaja ili tehnologije, kao što je upravljanje samim uređajem ili sigurnost. S vremenom takav je pristup počeo gubiti na popularnosti, a Windows Mobile počeo je gubiti teško stečeni tržišni udio. Sami uređaji i korisničko sučelje bili su usmjereni na robusnost, umjesto da pružaju korisniku prilagođen doživljaj [4].

U veljači 2010. godine Microsoft je na Mobile World Congressu otkrio potpuno novi sustav, sa promijenjenim imenom i novim pristupom mobilnoj tehnologiji.

Spomenuti operacijski sustav je nazvan Windows Phone 7, koji sa sobom donosi sasvim izmijenjenu filozofiju razvoja. To se odnosi kako na softverski tako i na hardverski dio platforme.

Microsoft je od same najave platforme odlučio uzeti konce čvrsto u svoje ruke i postaviti jasna ograničenja i pravila. Programeri i dizajneri aplikacija dobili su jasne



Slika 2. Osnovni WP7 izgled

upute kako bi trebalo izgledati sučelje jedne Windows Phone 7 aplikacije. Glede toga, korisničko sučelje je temeljeno na dizajnerskom principu kojeg su u Microsoftu nazivali *Metro*. Cilj tog principa je kreirati doživljaj koji je prilagođen svakom korisniku personaliziranim sadržajem (slika 2). Fokus je usmjeren na čistoću prikaza, brzinu i glatke animacije, integraciju hardvera i softvera te prezentiranje najvažnijeg sadržaja korisniku na njemu najprihvatljiviji i najlakši način. Microsoft je u svrhu postizanja tog cilja čak objavio i vodič [11] u kojem upisuje sve relevantne koncepte, pravila i „trikove“ koji mogu pomoći programerima i dizajnerima u razvoju takvih aplikacija.

S druge strane, i sami proizvođači uređaja koji će implementirati Windows Phone 7 operacijski sustav su dobili vrlo jasne upute u vezi minimalnih specifikacija koje moraju zadovoljavati. Kod ranijih izdanja (Windows Mobile) Microsoftovih mobilnih platformi minimalni hardverski zahtjevi su često bili preniski, kao i ograničenja prema ostalim zainteresiranim stranama (telekomunikacijskim tvrtkama, programerima i sl.), pa je kod spajanja svega toga u cjelinu krajnji proizvod često bio spor i nepouzdan, što se uglavnom pripisivalo samom operacijskom sustavu [5]. Microsoft je iskoristio priliku pri razvoju sasvim nove platforme i pažljivo isplanirao svaki aspekt koji može utjecati na zadovoljstvo krajnjeg korisnika. Neki od minimalnih hardverskih zahtjeva postavljenih pred proizvođače uređaja su sljedeći [4]:

- Procesor od minimalno 1GHz
- WVGA (480 x 800 px) prikaz
- MultiTouch sposobnost ekrana (min 4 točke)
- DirectX 9
- GPS, akcelerometar, kompas
- Hardverske kontrole: Back, Start i Search
- Mogućnost bežičnog povezivanja
- 256 MB RAM i 8 GB memorije

U budućnosti se predviđa i podrška za ekrane dimenzija razlučivosti 320 x 480 točkica [1], no u vrijeme objavljivanja platforme bila je moguća samo jedna rezolucija, što je uvelike olakšalo posao dizajnerima aplikacija.

Kao što je već nekoliko puta napomenuto, korisnički doživljaj je bio glavna nit vodilja kod razvoja Metro korisničkog sučelja. Dio te filozofije možemo vidjeti i na samom početnom ekranu uređaja. Umjesto klasičnog

Start izbornika, korisnik vidi „pločice“ (*engl. tiles*) preko kojih pokreće aplikacije. No to nisu samo statične slike koje vas vode do aplikacija. One se cijelo vrijeme ažuriraju prikazujući korisniku informacije koje su njemu zanimljive i korisne. Osim toga, klikom na neku od pločica pokreće se animacija u kojoj se ona neznatno povećava, dok u isto vrijeme ostale pločice nestaju, a navigacija vas vodi na ciljanu aplikaciju. Konzistentnost prikaza je također vrlo važna, tako da je kroz cijeli uređaj vođen taj isti dizajnerski princip. Takvi detalji su ono što se u Microsoftu konstantno ističe kao sjajan korisnički doživljaj.

4. ELEMENTI RAZVOJA APLIKACIJE

4.1 Nokia QT elementi aplikacije

Prilikom razvoja QT SmartBuy aplikacije bilo je potrebno napraviti nekoliko formi aplikacije sa pripadajućim kontrolama za pokretanje aplikacijske logike. Prva forma koja se prikazuje ima za zadatak pružanje osnovnog sučelja sa glavnim funkcijama programa te služi za pokretanje funkcije `startGPS()` koja dohvaća GPS podatke (geografsku širinu i dužinu) trenutne lokacije mobilnog telefona. Za implementaciju dohvaćanja GPS podataka korišten je QtMobility API.

Tijekom procesa dohvaćanja GPS podataka, na glavnom prozoru aplikacije bit će omogućeno korištenje samo jedne kontrole, gumba za izlaz iz programa, sve do trenutka kada se dohvate GPS podaci te se tada omogućava korištenje svih ostalih kontrola. Na taj način se sprječava korištenje aplikacije, odnosno pokretanje njenih funkcionalnosti koje se temelje između ostalog na podacima o geografskoj širini i dužini, ukoliko ti podaci nisu prisutni u memoriji aplikacije.

U isječku programskog koda (1) prikazana je funkcija koja se poziva u trenutku dobivanja GPS signala.

Kontrole šalju signale prilikom okidanja određenog događaja. Tako na primjer, gumb će poslati signal `clicked` u trenutku kada je kliknut. Kako bi iskoristili signal koji označava da su GPS podaci dostupni, korišteno je povezivanje tog signala na funkciju `positionUpdated`, kao što je prikazano u programskom kodu (2).

Osnovno pretraživanje odvija se pozivanjem forme pod nazivom poslovnice kojem se proslijeđuju parametri: kriterij pretraživanja, geografska širina i dužina trenutne lokacije te maksimalni broj zapisa koje je potrebno pronaći. Ova forma koristi QtWebKit modul, odnosno kontrolu QWebView u kojoj se prikazuju poslovnice koje sadrže proizvod za kojim je pokrenuto pretraživanje (kriterij pretraživanja) pozivanjem PHP skripte koja sadrži markere na Google Mapi kao pozicije poslovnica. Udaljenoj PHP skripti se kroz parametre šalju nazivi poslovnica, geografska širina i dužina.

Kako bi implementirali funkcionalnost dohvaćanja podataka o poslovnicama koje sadrže traženi proizvod koristili smo ASP.NET web servis. U formi za prikaz poslovnica je isto tako korištena funkcija za dohvaćanje GPS lokacije te je definirano kako će se prikaz poslovnica u GoogleMapi osvježiti ukoliko se pozicija korisnika aplikacije promijeni za više od 100 metara. Na ovaj način smo postigli ažuriranje podataka ukoliko korisnik aplikacije mijenja svoju poziciju (na primjer ukoliko korisnik hoda ili se vozi u automobilu).

```

void MainWindow::positionUpdated(QGeoPositionInfo geoPositionInfo)
{
    //provjera ispravnosti dohvaćene pozicije
    if (geoPositionInfo.isValid())
    {
        //dohvaćanje trenutne pozicije
        QGeoCoordinate geoCoordinate = geoPositionInfo.coordinate();
        qreal latitude = geoCoordinate.latitude();
        qreal longitude = geoCoordinate.longitude();

        //pretvaranje u tekstualni oblik
        QString GeoLatitude = (QString::number(latitude));
        QString GeoLongitude = (QString::number(longitude));
    }
}

```

Kôd 1. Dohvaćanje GPS lokacije

Kako bi se uspješno moglo implementirati brisanje markera na PHP skripti koja prikazuje poslovnice preko GoogleMap tehnologije, za ponovno dodavanje ažuriranih lokacija poslovnica, potrebno je u konstruktoru forme omogućiti pozivanje JavaScript funkcija kako je prikazano u programskom kodu (3).

Kao što je već spomenuto, korišteni su ASP.NET web servisi za dohvaćanje podataka o poslovnicama, proizvodima poslovnica te o detaljima proizvoda. Funkcija PokreniPretrazivanje poziva odgovarajući web servis sa pripadajućim parametrima kao što je prikazano u kodu (4).

Nakon što se preko web servisa dohvate podaci, poziva se funkcija getResponse, koja je registrirana uz signal responseReady, a koja omogućava prikaz dohvaćenih podataka u odgovarajućoj kontroli. Taj element funkcionalnosti je prikazan u programskom kodu (5).

Na sličan se način pokreću web servisi koji vraćaju popis akcija u blizini korisnika ove aplikacije, napredno pretraživanje, dohvaćanje podataka o proizvodima odabrane poslovnice te dohvaćanje podataka o odabranom proizvodu.

Kako bi se dohvaćeni podaci o poslovnicama ispravno prikazali u tablici (odnosno u kontroli za prikaz podataka u tablici, QTableView), korištena je klasa QStandardItemModel kojoj se podaci dodaju kao u tablicu. Na ovaj način, podaci se vrlo lako mogu prikazati u kontroli QTableView te se može pristupiti poziciji odabranog podatka, što je važno za pokretanje forme za prikaz detalja proizvoda koji između ostalih parametara konstruktora zahtijeva i redni broj poslovnice u kojoj se traži proizvod. Prilikom dohvaćanja podataka sa web servisa i prikaza na formi, postojao je problem oko prikazivanja hrvatskih znakova što je riješeno korištenjem funkcije QString::fromUTF8, kao što je

```

QObject::connect(locationDataSource, SIGNAL(positionUpdated(QGeoPositionInfo)),
    this, SLOT(positionUpdatedHandler(QGeoPositionInfo)));

```

Kôd 2. Povezivanje signala i slotova

```

ui->webView->settings()->setAttribute(QWebSettings::JavascriptEnabled,true);
ui->webView->settings()->setAttribute(QWebSettings::LocalContentCanAccessRemoteUrls,
    true);

```

Kôd 3. Omogućavanje JavaScript funkcionalnosti

```

void Poslovnice::PokreniPretrazivanje()
{
    QtSoapMessage request;
    QtSoapHttpTransport http;

    connect(&http, SIGNAL(responseReady()), this, SLOT(getResponse()));

    request.setMethod("VratiPopisPoslovnica","http://tempuri.org/");
    request.addMethodArgument("latitude", "", GeoLatitude);
    request.addMethodArgument("longitude", "", GeoLongitude);
    request.addMethodArgument("kriterijPretrazivanja", "", kriterij);
    request.addMethodArgument("brojZapisa", "", broj);

    http.setHost("smartbuy-project.com");
    http.setAction("http://tempuri.org/VratiPopisPoslovnica");
    http.submitRequest(request, "/servisi/PoslovniceWebServis.asmx");
}

```

Kôd 4. Poziv web servisa

```

void Poslovnice::getResponse()
{
    const QtSoapMessage &message = http.getResponse();
    if (message.isFault())
    {
        qDebug("Error: %s",
            message.faultString().value().toString().toLatin1().constData());
    }
    else
    {
        const QtSoapType &response = message.returnValue();

        // postavljanje dohvacenih podataka u odgovarajucu kontrolu
    }
}

```

Kôd 5. Dohvaćanje podataka sa web servisa

prikazano u isječku programskog koda (6).

Razvoj QT SmartBuy aplikacije se temeljio na razvoju za mobilni uređaj Nokia N97 mini, koji nam je služio i kao platforma za testiranje i daljnji razvoj, ali se mobilna aplikacija može izvršavati na drugim mobilnim telefonima koji sadrže Symbian operacijski sustav.

4.2 WP7 elementi aplikacije

Microsoft je 2006. godine, sa izlaskom .NET Frameworka 3.0 predstavio novi koncept u razvoju aplikacija. Izdan je Windows Presentation Foundation (WPF), novi grafički podsustav za prikaz korisničkih sučelja u aplikacijama. Jedna od ideja bila je jasno razdvojiti razvoj poslovne logike od razvoja korisničkog sučelja. Programeri poslovnu logiku najčešće razvijaju u programskom jeziku C#, dok je za korisničko sučelje Microsoft razvio novi deklarativni jezik – XAML (*engl. Extensible Application Markup Language*, čita se „zammeł“). XAML se zasniva na XML (*engl. Extensible Markup Language*) jeziku i hijerarhijski prikazuje strukturu korisničkog sučelja. XAML je moćan jezik koji uz primarnu zadaću definiranja prezentacijskog sloja aplikacije ima i druge važne uloge i podržava nove koncepte. Pomoću XAML-a mogu se jednostavno definirati stilovi i predložci kontrola korisničkog sučelja, omogućuje povezivanje podataka (*engl. data binding*), animacije, a aplikacije se na kraju izvršavaju kao upravljani kôd (*engl. managed code*), preko Common Language Runtime (CLR) sloja, što omogućuje laku integraciju i povezivanje sa ostatkom platforme. U 2007. godini izdana je prva verzija Silverlight projekta, koji je podskup WPF-a dizajniran za izvođenje u web okruženju.

Vođen tim konceptima, Microsoft zapravo nije „izmislio“ nove tehnologije specijalno za Windows Phone 7

platformu. Jednostavno su primijenili postojeće razvojne okvire (*engl. framework*) [4]. Uključene su tehnologije .NET, Silverlight i XNA (korišten kod razvoja igara za Xbox platformu), a to u prijevodu znači da se u najviše scenarija kod razvoja Phone 7 aplikacija XAML-om definira korisničko sučelje, a poslovna logika se razvija pomoću C# programskog jezika. Loša strana upravljanog koda je što je on sporiji u odnosu na urođeni (*engl. native*) kôd, no to je jedan od razloga zašto je Microsoft postavio relativno visoke hardverske zahtjeve pred Windows Phone 7 mobilne uređaje.

Dvije najčešće spominjane značajke XAML-a u kontekstu razvoja za Phone 7 uređaje su predložci (*engl. templates*) i povezivanje s podacima. Primjer toga možemo vidjeti u XAML kodu (7).

Ovo je vrlo jednostavan primjer predložka, koji omogućuje dizajneru/programeru aplikacije da potpuno promijeni izgled neke komponente korisničkog sučelja bez da mijenja njeno ponašanje. U ovom primjeru umjesto standardnog izgleda kontrole Pushpin, koja se koristi kod označavanja lokacija kod prikaza na geolokacijskim mapama, koristimo vlastiti izgled koji uključuje jednostavnu crnu granicu sa zaobljenim rubovima (CornerRadius), unutar koje se prikazuje tekst (pomoću kontrole TextBlock). Vidimo i jednostavan primjer povezivanja podataka, koji je pridružen atributu Text kontrole TextBlock. Povezivanje nam omogućuje da deklarativno uspostavimo vezu između elementa korisničkog sučelja i nekog izvora podataka. Ovaj je koncept uistinu koristan i pruža nov način pristupu razvoja aplikacija sa gledišta povezivanja sučelja i pozadinskih podataka. Izvor podataka može se eksplicitno specificirati, a ako to nije slučaj, izvor se traži tako da se kreće prema hijerarhijski višim razinama sučelja (izvor se nasljeđuje od roditelja) sve dok se ne pronađe [2]. U primjeru iznad, povezivanje

```
QString naziv = QString::fromUtf8(naziv_proizvoda);
```

Kôd 6. Korištenje hrvatskih dijakritičkih znakova u Qt-u

```

<ControlTemplate x:Key="myPushPin" TargetType="my:Pushpin">
    <Border BorderBrush="Black" BorderThickness="2" Width="40" Height="40" CornerRadius="5">
        <TextBlock Text="{TemplateBinding Content}"></TextBlock>
    </Border>
</ControlTemplate>

```

Kôd 7. Korištenje predložaka pri povezivanju s podacima

```
<my:Pushpin x:Name="myPin" Content="X" Template="{StaticResource myPushPin}" />
myPin.Location = new GeoCoordinate(45.8, 15.9);
myPin.Content = "X";
```

Kôd 8. Izmjena sadržaja i svojstava kontrole

kao izvor podataka ima atribut Content, koji se može pronaći u pripadajućoj instanci izvornog tipa predloška (kontrola Pushpin). Nakon uspostavljanja te veze, svaki put kada se promijeni vrijednost Content atributa, promijenit će se i tekst pripadne instance kontrole. Sjajno svojstvo XAML-a je što je sve promjene moguće vršiti i u „pozadinskom“ C# kodu, tako da je između ostalog moguće izgled, sadržaj i svojstva kontrola mijenjati i tijekom rada same aplikacije.

Primjer možemo pokazati na upotrebi gornje personalizirane kontrole PushPin. Instancirat ćemo ga u samom XAML-u, a lokaciju na mapi i sadržaj teksta promijenit ćemo u pozadinskom C# kodu (8).

Predložak za ovaj primjer je nadograđen upotrebom plave boje za granicu kontrole, kao i gradijenta za pozadinu same kontrole. XAML kôd je u ovom slučaju dosta dulji pa nije stavljen radi preglednosti, no funkcionalnost i logika ostaje ista. Tu također možemo vidjeti koliko je jednostavno mijenjati dizajn kontrola i aplikacije bez da to utječe na ostatak aplikacije (npr. poslovnu logiku). U C# kodu je lokacija kontrole na mapi postavljena na područje Zagreba (pomoću geografske širine i dužine), a sadržaj (Content) kontrole na slovo X. Pošto je u predlošku definirano da je tekst kontrole povezan sa tim sadržajem, on se također automatski

čiji se dosadašnji operacijski sustav, Symbian, zadnjih godina pokazao inferiornim u odnosu na konkurenciju, i zbog čega Nokia već duže vrijeme ubrzano gubi tržišni udio na području tzv. pametnih mobilnih uređaja (engl. *Smartphone*) [6].

1.3 Usporedba QT i WP7 razvoja

Qt skup alata za razvoj programa (engl. *Qt Software Development Kit*) programerima olakšava razvoj aplikacija na način da im omogućava da jednom napisanu aplikaciju isporuče (engl. *Deploy*) na više platformi. Ovaj skup alata namijenjen je pisanju desktop aplikacija za Windows, Linux i Mac OS X, te za pisanje mobilnih aplikacija namijenjenih Symbian i Maemo platformi [10]. Qt također omogućuje programerima da im aplikacije budu spremne za nadolazeću MeeGo platformu [9].

Da bi se razvila aplikacija u Qt-u, potrebno je poznavati C programski jezik, dok za razvijanje aplikacije za Windows Phone 7 treba znati programirati u C#. Da bi bilo što lakše razvijati aplikacije, uređivač koda u Qt SDK-u (Carbide) podržava izradu grafičkog sučelja po principu „drag & drop“. To znači da raspolaže sa već gotovim kontrolama koje jednostavno možemo odabrati u panelu sa kontrolama i „odvući“ na formu naše aplikacije. Baš u tom dijelu su Qt i WP7 slični zbog toga što Visual Studio (razvojno okruženje za WP7 aplikacije) također ima panele sa ponuđenim gotovim kontrolama za izradu grafičkog sučelja aplikacije.

Osim navedenog, Qt i Windows Phone 7 su slični po tome što imaju odvojene datoteke za dizajn grafičkog sučelja od ostalog dijela aplikacije. U oba slučaja radi se o XML formatu. No, s druge strane tu postoji i velika razlika, a to je što Qt ne omogućava uređivanje grafičkog sučelja direktnim uređivanjem XML-a, dok je to kod Windows Phonea moguće. Kao što smo već spomenuli, za Windows Phone koristi se posebna verzija XML jezika, a to je XAML.

Izgled sučelja u Qt-u se može dodatno prilagoditi koristeći CSS stilove (engl. *Cascading Style Sheets*). Microsoft ovdje ima puno bolju podršku za prilagođavanje sučelja, te je ono puno jednostavnije zbog toga što su XAML datoteke, u kojima se definira sučelje, jednostavne za uređivati, pogotovo uz pomoć dodatka IntelliSense koji je ugrađen u Visual Studio. Microsoft također nudi poseban alat za uređivanje grafičkog sučelja aplikacija. Radi se o Expression Blendu. U njemu je moguće jednostavno otvoriti već prije kreiran Visual Studio projekt, urediti grafičko sučelje po svojoj želji, spremati promjene i tada u Visual Studiu nastaviti sa programiranjem funkcionalnosti aplikacije.

Snaga Qt-a leži u njegovim aplikacijskim programskim sučeljima (engl. *Application Program Interface – API*), što nam omogućava da pišemo složenije aplikacije sa manje koda, a to dovodi i do veće brzine razvoja aplikacije. Najkorisniji paket API-a za razvoj mobilnih aplikacija su Qt Mobility API-ji. Tu spadaju programska sučelja za dohvaćanje podataka sa GPS-a, slanje SMS poruka i slično. Microsoft je ovdje za WP7 odlučio razviti tzv. servise za slične namjene, pa je tako moguće koristiti servise za akcelerometar i lokaciju (geolokacija) [1; 80]. Vrijedi spomenuti da se npr. poziv API-a za lokaciju kod Qt-a i poziv servisa za lokaciju kod Windows



Slika 3. Prikaz korisnički definirane kontrole

mijenja. Predložak se u aplikaciji definira kao resurs, pa je on kontroli pridružen ključnom riječi *StaticResource*.

Vidjeli smo najvažnije elemente razvoja za Windows Phone 7 platformu, koja je uistinu široko područje. Kao primjer možemo spomenuti XNA koji se koristi za izradu igara na Phone 7 mobilnim uređajima, a koji je još jedno široko područje za istraživanje. Tržište uređaja kao i aplikacija za Phone 7 je još uvijek vrlo mlado ali je itekako pokazalo potencijala, tako da se može smatrati novim „velikim igračem“ na području mobilnih tehnologija, uz već dokazane Android i iOS sustave. Tome u prilog ide i objavljena suradnja Microsofta s još uvijek najvećim proizvođačem mobilnih uređaja, Nokiom,

Phone 7 zapravo svodi na isto: pokreće se dretva koja konstantno provjerava trenutnu geolokaciju mobitela.

Vidjeli smo da postoji mnogo sličnosti i razlika između Qt-a i WP7, no važno je uočiti da su glavne sličnosti u tome što je grafičko sučelje odvojeno od programskog koda, te da obje platforme (Qt i Windows Phone 7) koriste dodatke (API-e ili servise) koji omogućuju jednostavnije korištenje raznih funkcionalnosti na mobilnim uređajima, kao što je dohvaćanje geolokacije uređaja.

5. ZAKLJUČAK

Razvojem ovih aplikacija mnogo smo naučili o Qt-u i Windows Phone 7. Qt ima dobru podršku za mobilne platforme u obliku raznih API-a. Konstantnom nadogradnjom, svakom novom verzijom dostupno je nekoliko novih API-a, pa je tako od nedavno moguće koristiti API pod nazivom „maps and navigation“ (*hrv. mape i navigacija*). On se zasniva na dodacima (*engl. plugins*) pomoću kojih se komunicira sa raznim pružateljima usluga za mape, geokodiranje i određivanje puta [8]. Prije nekoliko mjeseci tog API-a nije bilo, pa su se mape morale koristiti na druge načine. Danas se,

zahvaljujući ovoj nadogradnji, lakše mogu razvijati aplikacije koje koriste mape i geolociranje. Isti je slučaj i sa drugim API-ima, te se daje naslutiti da će ovdje biti još puno novosti.

Windows Phone 7 je relativno nova platforma, pogotovo ako je uspoređujemo sa Qt-om, odnosno Symbianom. No, bez obzira na to, već sad se vidi da je ona na dobrom putu da postane jedna od omiljenih platforma developera, a najviše zbog načina na koji je odvojeno grafičko sučelje od programskog koda aplikacije. Također jedan od razloga je i taj što se za razvoj aplikacija koristi programski jezik C# koji je već vrlo dobro poznat među programerima.

Na kraju, želimo se osvrnuti i na sâm trend porasta broja mobilnih aplikacija na tržištu. Smatramo da je ovo područje vrlo perspektivno i da pruža prilično velike mogućnosti. Najvažnije od svega je to što sada svi programeri koji razvijaju mobilne aplikacije i objavljuju ih na nekoj od trgovina (Ovi Store, App Store, Android Market, Windows Marketplace...) imaju jednake šanse za uspjeh. Sve što je potrebno jest znati dobro programirati i uz malo volje i inovativnosti može se napraviti kvalitetna aplikacija koju će koristiti velik broj korisnika.

Literatura:

- 1 Charles Petzold: Programming Windows Phone 7, Microsoft Press, Washington, 2010.
- 2 Christopher Bennage, Rob Eisenberg: Getting Started with Windows Presentation Foundation, Dzone, 2009.
- 3 Faster, More Cost-Effective Development Using the Qt Cross-Platform Application & UI Framework, Qt – Cross-platform application and UI framework, 2009. Izvor: <http://qt.nokia.com/files/pdf/faster-more-cost-effective-development-using-the-qt-cross-platform-application-ui-framework> (učitano, 14.4.2011.)
- 4 Henry Lee, Eugene Chuvyrov: Beginning Windows Phone 7 Development, Apress, 2010.
- 5 Nick Randolph, Christopher Fairbairn: Professional Windows Phone 7 Application Development, Wiley Publishing, 2011.
- 6 Nokia and Microsoft Announce Plans for a Broad Strategic Partnership to Build a New Global Mobile Ecosystem, Microsoft Corporation, 2011. Izvor: <http://www.microsoft.com/presspass/press/2011/feb11/02-11partnership.mspx> (učitano, 5.5.2011.)
- 7 Nokia QT Whitepaper, Nokia Corporation, 2009. Izvor: <http://qt.nokia.com/files/pdf/qt-4.6-whitepaper> (učitano, 13.4.2011.)
- 8 Qt Mobility Project Reference Documentation, Qt – Cross-platform application and UI framework, 2010. Izvor: <http://doc.qt.nokia.com/qtmobility-1.1.3/location-overview.html> (učitano, 06.05.2011.)
- 9 Qt Overview, Forum Nokia, 2011. Izvor: <http://www.forum.nokia.com/Develop/Qt/> (učitano, 06.05.2011.)
- 10 Qt Reference Documentation: Qt SDK 1.1, Qt Online Reference Documentation, 2011. Izvor: <http://doc.qt.nokia.com/sdk-1.1/index.html> (učitano, 06.05.2011.)
- 11 UI Design and Interaction Guide for Windows Phone 7, Microsoft Corporation, 2010. Izvor: <http://go.microsoft.com/fwlink/?LinkID=183218> (učitano, 14.4.2011.)

Podaci o autorima:

Zlatko Stapić, mag. inf.

tel: +385 42 390 853

fax: +385 42 213 413

e-mail: zlatko.stapic@foi.hr

Zlatko Stapić, mag. inf. je od 2006. godine asistent na Katedri za razvoj informacijskih sustava na Fakultetu organizacije i informatike u Varaždinu, te polaznik poslijediplomskog doktorskog studija Informacijske znanosti na istom fakultetu. Njegova nastavna aktivnost je prvenstveno usmjerena na kolegije koji se odnose na programsko inženjerstvo, analizu i razvoj programa, modeliranje poslovnih procesa i razvoj informacijskih sustava, te značajne napore ulaže u radu sa studentima za što je dobio i posebna priznanja.

Iz znanstvenog i stručnog rada treba izdvojiti višegodišnje voditeljstvo stručnih projekata razvoja programskih proizvoda i sudjelovanje na različitim stručnim i znanstvenim projektima iz područja razvoja, unapređenja poslovnih procesa, projektnog menadžmenta i slično.

U posljednje vrijeme se intenzivno bavi razvojem aplikacija za mobilne uređaje, što je i predmet njegovog istraživanja u okviru doktorske disertacije, a osobito je vrijedno istaknuti da razvija za skoro sve platforme, uključujući između ostalog Android, Symbian te Windows Phone 7. Zlatkov detaljniji životopis, s popisom svih radova, projekata i nagrada, te drugih važnih podataka može se pronaći na njegovoj osobnoj web stranici, na <http://www.foi.hr/djelatnici/zlatko.stapic>.

Zlatko Stapić is a young researcher and teaching assistant at the Faculty of Organization and Informatics working at the Information systems development department. His main areas of interests include classic and agile software engineering methodologies, software and information systems development, business processes modeling and others.

Josip Vincek, univ. bacc. inf.

e-mail: josip.vincek@foi.hr

Josip Vincek završio je Elektrostrojarsku školu u Varaždinu 2007. godine, smjer tehničar za računalstvo. Iste godine upisao je prvu godinu preddiplomskog studija na Fakultetu organizacije i informatike u Varaždinu kojeg je završio 2010. godine stekavši tako titulu prvostupnika informatike. Iste godine na matičnom fakultetu upisao je i diplomski studij, smjer Informacijsko i programsko inženjerstvo. Za rad na projektu „Određivanje karakterističnih otisaka slike dlana“ dobio je dekanovu nagradu. Područja interesa su mu razvoj aplikacija za stolne, web i mobilne platforme kao i praćenje modernih tehnoloških trendova u razvoju takvih sustava.

Zoran Šaško, univ. bacc. inf.

e-mail: zoran.sasko@foi.hr

Zoran Šaško je 2010. godine završio sveučilišni preddiplomski studij na Fakultetu organizacije i informatike u Varaždinu, smjer Informacijski sustavi. Iste je godine upisao diplomski studij smjer Informacijsko i programsko inženjerstvo. Kroz duži broj godina se bavi programiranjem web i desktop aplikacija, pa je iz afiniteta prema razvoju programskih rješenja započela i suradnja s kolegama na razvoju SmartBuy mobilne aplikacije. Osim programiranja za web i mobilne telefone, područje interesa mu je i internet marketing, odnosno promocija proizvoda i usluga preko interneta.

Miroslav Zver, univ. bacc. inf.

e-mail: miroslav.zver@foi.hr

Miroslav Zver je 2010. godine na Fakultetu organizacije i informatike u Varaždinu završio sveučilišni preddiplomski studij smjer Informacijski sustavi i stekao titulu prvostupnika informatike. Iste godine na FOI-u je upisao diplomski studij smjer Informacijsko i programsko inženjerstvo. Sa kolegama Šaškom i Vincekom sudjelovao je na natječaju Nokia Symbian Challenge, a iz te suradnje nastala je ideja za aplikaciju predstavljenu u ovom radu. U posljednje vrijeme bavi se izradom mobilnih aplikacija, ali zanima ga i web programiranje. Osim aktivnosti vezanih uz fakultet, unutar udruge LepoMAN (*engl. Lepoglava Metropolitan Area Network*) bavi se mrežnim tehnologijama.

Fakultet organizacije i informatike

Pavlinska 2

42000 Varaždin