

IX. VIZUELNO PROGRAMIRANJE I .NET

Sadržaj prethodnih poglavlja se prije svega odnosio na osnove programiranja proceduralnim jezicima, uz praktičnu primjenu kroz Liberty BASIC, kao jedan od proceduralnih programskih jezika. Savremene aplikacije podrazumijevaju prije svega grafički korisnički interfejs, čitavu lepezu funkcionalnosti, komunikaciju s nekom bazom podataka, pa čak i web orijentaciju. Većina navedenih zahtjeva se može zadovoljiti korištenjem proceduralnih programskih jezika, ali se postavlja pitanje isplativosti sa aspekta vremena i ljudskih resursa, potrebnih za razvoj aplikacija proceduralnim jezicima.

Zahtjevi za dinamičnim razvojem aplikacija, posebno modula poslovnih informacionih sistema, su doveli do pojave alata za brzi razvoj aplikacija. Ovi alati nisu bili samo kompajleri i editori koda, nego **čitava razvojna okruženja** koja su sa sobom donosila i promjene u načinu pristupa razvoju aplikacija, njihovom projektovanju, dizajniranju, implementaciji i održavanju. U posljednjim verzijama ovakvih alata, dostupnih na tržištu, spominju se "okviri" za razvoj aplikacija, odnosno setovi alata i pravila za njihovo korištenje, koji definišu način pristupa razvoju i održavanju aplikacija.

1. Brzi razvoj aplikacija

Brzi razvoj aplikacija ili RAD (engl. Rapid Application Development) je pojam koji se u početku koristio za označavanje iterativnog procesa razvoja aplikacija uz korištenje prototipova, a u novije vrijeme označava pristup razvoju aplikacija, koji omogućava brzi razvoj uz korištenje što više općih, ranije razvijenih dijelova koda. Ovo praktično znači da se programerima pružaju mnogo veće mogućnosti za reupotrebu programskog koda i na taj način se skraćuje vrijeme potrebno za razvoj aplikacija, ili njihovih modula.

S obzirom na široku rasprostranjenost operativnih sistema koji se oslanjaju na grafički korisnički interfejs, korisnici od današnjih aplikacija zahtijevaju prije svega jednostavan i upotrebljiv grafički interfejs, ali i implementaciju svih neophodnih funkcionalnosti unutar aplikacija koje se razvijaju. S druge strane, menadžeri, koji u konačnici odlučuju o korištenju poslovnih aplikacija u kompanijama, zahtijevaju što manje troškove i što skorije uvođenje novih aplikativnih rješenja. Ovakvi zahtjevi, vezani za razvoj aplikacija, dovode programere u nezavidan položaj, s obzirom da za funkcionalan grafički interfejs razvoj proceduralnim jezicima može biti dugotrajan, uz visoke zahtjeve za ljudskim resursima.

Svi navedeni razlozi opravdavaju kreiranje razvojnih okruženja koja omogućavaju brzi razvoj aplikacija, ili njihovih modula, uz olakšano kreiranje grafičkog interfejsa i programske logike koja implementira sve neophodne funkcionalnosti. Kroz korištenje ovakvih okruženja, programeri dobivaju alate za kreiranje grafičkog korisničkog interfejsa jednostavnim povlačenjem komponenti na radnu površinu, koja će u konačnici predstavljati formu programskog interfejsa prema korisniku. Na forme se stavljaju dugmad, text-polja, labela, liste za izbor i druge komponente koje zajednički čine interfejs prema korisniku. Na taj način se svo vrijeme i trud potreban za programiranje ovih komponenti proceduralnim jezicima smanjuje na najmanju moguću mjeru, a programeru daje više vremena i mogućnosti da kvalitetno kreira programski kod koji se inicira akcijama korisnika na interfejsu, kao što su klik na dugme, upis teksta u tekstualno polje, ili izbor opcije u listi za izbor.

RAD razvojna okruženja na taj način ubrzavaju razvojni proces, omogućavaju kreiranje kvalitetnijih aplikacija i koncentrisanje napora na implementaciju programske logike, umjesto interfejsa. Osim toga, RAD alati razdvajaju elemente interfejsa od koda koji implementira programsku logiku i tako omogućavaju paralelan razvoj interfejsa s jedne i programske logike s druge strane, ukoliko je to fizički moguće.

Tipični predstavnici ovakvih razvojnih okruženja su Microsoft .NET, Oracle JDeveloper, CodeGear Delphi, ili Oracle Forms. Svako od ovih okruženja donosi svoje prednosti i nedostatke i ovisno o tome kakve aplikacije će se razvijati može se preferirati jedno okruženje u odnosu na druga rješenja.

2. Microsoft .NET

.NET je skup tehnologija razvijenih od strane kompanije Microsoft za brzi razvoj različitih tipova aplikacija. Ovi tipovi uključuju:

- web bazirane aplikacije,
- windows aplikacije zasnovane na formama i grafičkom korisničkom interfejsu,
- biblioteke programskog koda,
- konzolne aplikacije,
- web servise, itd.

Može se primjetiti da je skup aplikacija koje se mogu razvijati primjenom .NETa raznolik. Da bi se ovakav razvoj omogućio, potrebno je da postoji razvojni okvir koji sadrži biblioteke programskog koda za olakšan razvoj, te pravila pristupa razvoju pojedinih tipova aplikacija. Ovakav razvojni okvir se u .NET svijetu naziva .NET SDK (Software Development Kit). Drugim riječima, .NET SDK je skup biblioteka i klasa koje implementiraju najviše korištene programske funkcije i procedure, te elemente korisničkog interfejsa.

SDK sam po sebi omogućava programiranje, ali ne predstavlja alat za razvoj. Stoga .NET donosi i alat za razvoj koji se naziva Visual Studio .NET i koji predstavlja grafički IDE (Integrated Development Environment) za razvoj aplikacija na osnovu .NET SDK, koji se nalazi u osnovi Visual Studia.

Neke od karakteristika Visual Studia su da omogućava povlačenje komponenti u razvoju grafičkog interfejsa, daje prijedloge pri kodiranju programskog koda, ima odličnu podršku za uklanjanje grešaka u kodu, omogućava integraciju sa softverima za verzioniranje koda i olakšava kontrolu velikih razvojnih projekata. Ovo su samo neke od osnovnih prednosti Visual Studia kao razvojnog okruženja. Osim ovih prednosti, treba napomenuti da Visual Studio omogućava i razvoj aplikacija za prenosne platforme, kao što su mobilni i PDA (Personal Digital Assistant) uređaji.

Visual Studio kao softver ima nekoliko verzija, ovisno o nivou korištenja i programskom jeziku koji će se koristiti u razvoju aplikacija. Tako na primjer, postoje: Professional, Enterprise Developer i Enterprise Architect verzije, kao i Visual Basic .NET Standard Edition, Visual C# Standard Edition ili Visual C++ .NET Standard.

Sve verzije Visual Studia omogućavaju da se programeri s različitim predznanjem koriste istim setom alata za razvoj i na taj način se minimizira neophodna edukacija programera za korištenje novog alata za razvoj i nove filozofije razvoja uopće.

U prethodnim poglavljima dati su primjeri kreiranja windows aplikacija s grafičkim korisničkim interfejsom, na bazi primjene Liberty BASICa kao programskog jezika za razvoj. Pokazalo se da za razvoj samo jednog ekrana, s nekoliko dugmadi, ili tekstualnih polja treba napisati minimalno onoliko naredbi programskog jezika koliko ima komponenti na interfejsu, uz dodatne naredbe za otvaranje i pozicioniranje prozora. Koristeći se Visual Studiem, grafički interfejs se može razviti bez pisanja programskog koda, i to samo povlačenjem komponenti u budući prozor i njihovim smještanjem uz pomoć miša na odgovarajuće lokacije. Istina je da se programski kod za implementaciju funkcionalnosti iniciranih akcijama na interfejsu (klik na dugme i slično) i u Visual Studiu mora isprogramirati u respektivnom programskom jeziku, ali je broj linija koda potreban za to mnogo manji nego kod, na primjer, Liberty BASICa. Osim toga akcije klika na dugme, zatvaranja prozora, promjene veličine prozora i slično su u Visual Studiu već razvijene i dio su komponenti kojima se koristimo u izgradnji grafičkog interfejsa.

Ovo su osnovne prednosti .NET okruženja u odnosu na standardni pristup razvoju aplikacija na bazi primjene proceduralnih programskih jezika.

Dodatne prednosti su: olakšan rad s bazama podataka, fajl sistemom i uopće ulazno/izlaznim uređajima, što kod poslovnih aplikacija predstavlja kritičan faktor u razvoju.

Općenito se može reći da .NET i slična okruženja predstavljaju danas de facto standard u razvoju aplikacija, ali su previše kompleksna za programere početnike, koji se ipak prvo moraju upoznati s proceduralnim programskim jezicima i načinom razmišljanja jednog programera.