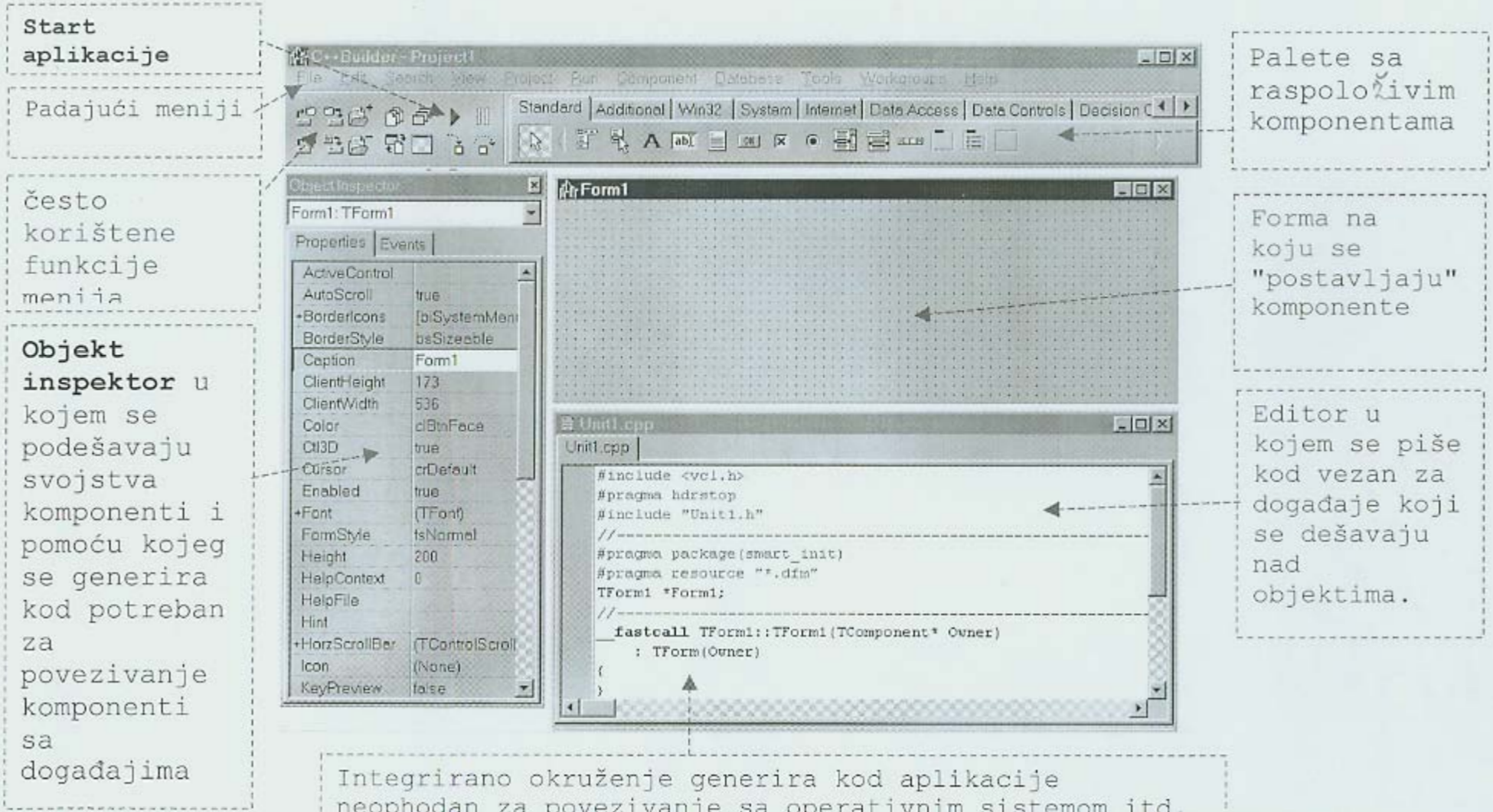
1. Integrirano okruženje Borland Builder v. 3.0

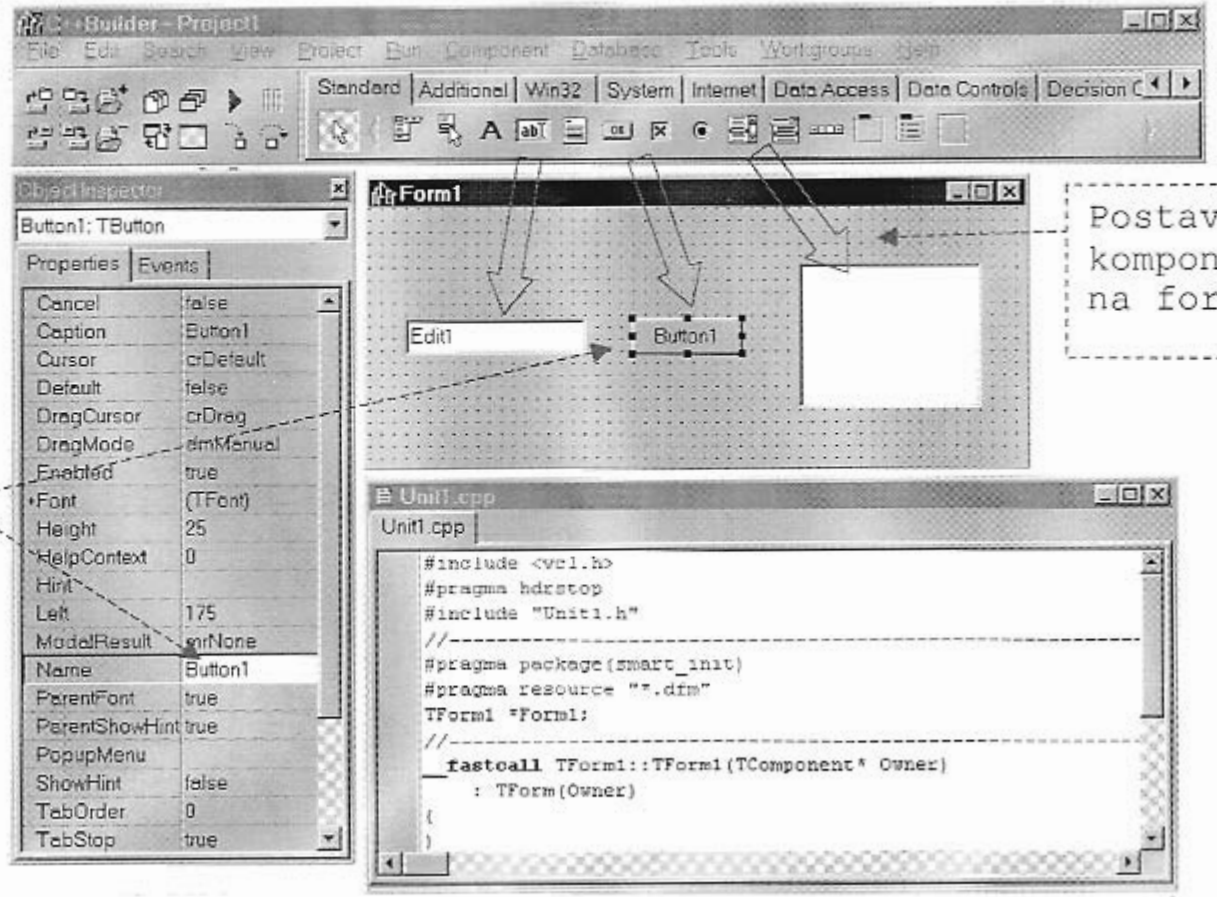


2. Kreiranje aplikacije

2.1. Postavljanje komponenti na formu i podešavanje svojstava komponenti

U objekt inspektoru se podešavaju svojstva trenutno odabrane (selekttirane) komponente.

Svojstva koja se mogu podešavati samo u doba kreiranja aplikacije nazivaju se **statička svojstva.** Svojstva koja se mogu mjenjati i tijekom izvršavanja aplikacije nazivaju se **dinamička svostva.**

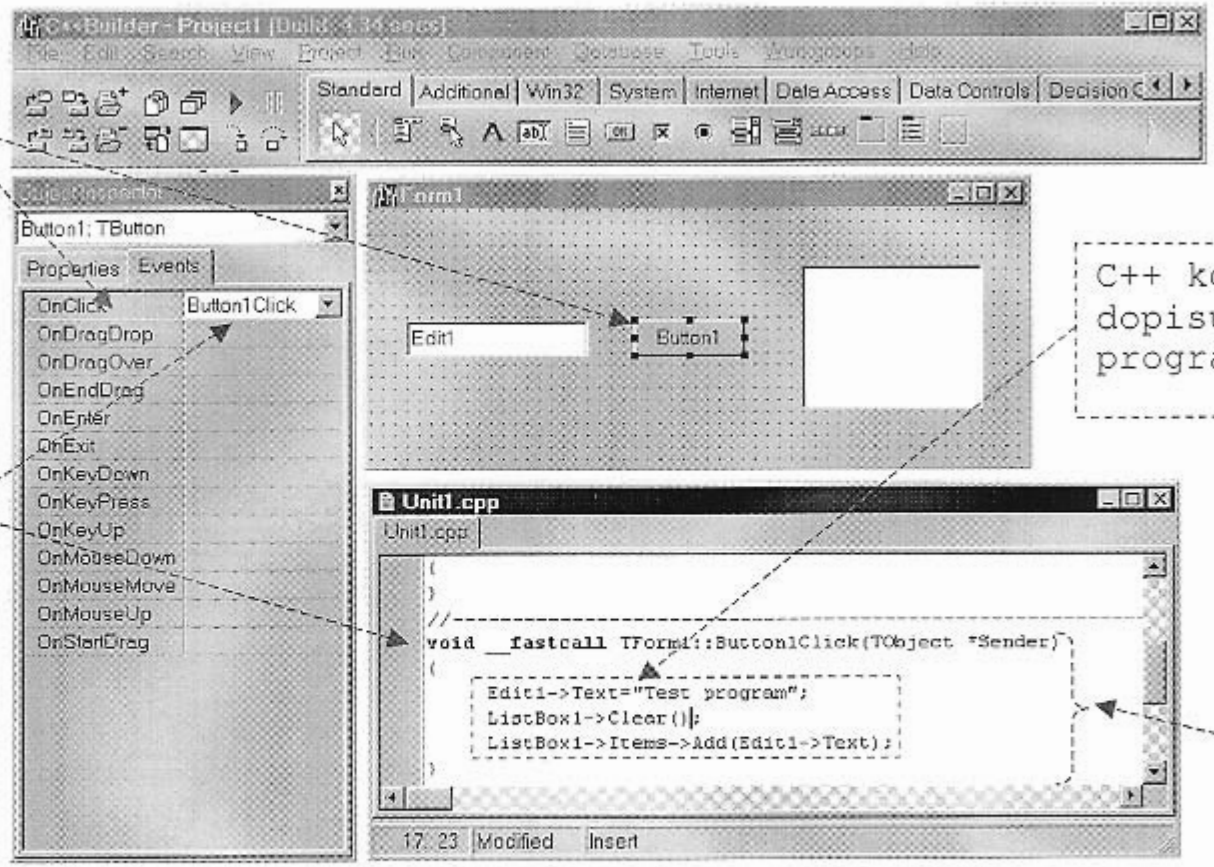


Postavljanj komponente na formu

2.2. Programiranje događaja, podešavanje svojstava i pozivanje metoda komponenti

U objekt inspektoru se odabiraju događaji na koje treba da "reagira" selektirana komponenta.

Dvoklikom nad praznim poljem desno od opisa mogućeg događaja nad komponentom, integrirano okruženje generira praznu funkciju povezanu sa mogućim događajem nad objektom



C++ kod koji dopisuje programer

Definicija funkcije za podrazumjevani događaj nad komponentom se može generirati i dvoklikom nad komponentom.

Funkcija koja će se izvršiti kad se desi događaj nad komponentom

Programiranje događaja:

Na prethodnoj slici je prikazano kreiranje aplikacije u kojoj je događaj "klik" mišem (Click) na komponentu taster (Button1), koji se nalazi na formi (Form1), povezan sa funkcijom koju je generiralo integrirano okruženje:

```
//---------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{

}
//---------------------------------------------------------------
```

U prazno tijelo funkcije **TForm::Button1Click(TObject *Sender)** programer upisuje svoj kod koji će se izvršiti svaki put kad se desi događaj "klik" (Click) nad objektom taster (Button1):

```
//---------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
     Edit1->Text="Test program";         // promjena teksta u komponenti Edit1
     ListBox1->Clear();                  // brisanje sadrzaja u ListBox1
     ListBox1->Items->Add(Edit1->Text); // upisivanje u ListBox1 teksta iz Edit1
}
//---------------------------------------------------------------
```

Podešavanje svojstava:

Komponente imaju statička i dinamička svojstva.
Statička svojstva se pode{avaju samo u objekt inspektoru za vrijeme kreiranja aplikacije.
Dinamička svojstva se podešavaju i u objekt inspektoru za vrijeme kreiranja aplikacije i pozivanjem odgovarajućih naredbi za dodjelu vrijednosti dinamičkim svojstvima tijekom izvršenja aplikacije.

Primjer podešavanja **dinamičkih svojstva** komponente **Edit1** je promjena teksta upisanog u o svojstvo za vrijeme izvršavanja aplikacije:

```
Edit1->Text="Test program";          // promjena teksta u komponenti Edit1
```

Pozivanje metoda:

Funkcije ugrađene u komponente i koje pri pozivanju izvrše neku akciju nad komponentom, nazivaju se metodama.

Primjer pozivanja metoda koje rade nad objektom **ListBox1** i koje brišu sadržaj te komponente i dopisuju novi sadržaj u nju:

```
ListBox1->Clear();                          // brisanje sadrzaja u ListBox1
ListBox1->Items->Add(Edit1->Text);      // upisivanje u ListBox1 teksta iz Edit1
```

2.3. Brisanje komponente i koda pridruženog komponenti

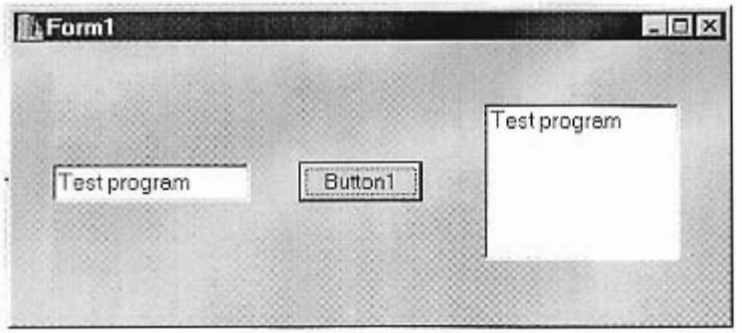
Komponenta se briše tako što se selektira, a zatim se odradi akcija **Cut** ili **Delete** iz menija **Edit,** ili tako što se selektira komponenta a zatim pritisne dugme **Ctrl+Del** ili samo dugme **Del**.

Kod koji je programer upisao u funkciju koju je generiralo integrirano okruženje za obrisanu komponentu, treba obrisati.
NE SMIJE se brisati kod koji programer nije sam napisao!!!
Taj kod će integrirano okruženje samo obrisati. U protivnom, integrisano okruženje će prijaviti greške...

2.4. Pokretanje aplikacije

Aplikacija se starta klikom na ikonicu **Run**, pritiskom na taster **F9** ili odabirom opcije **Run** iz **Run** padajućeg menija.

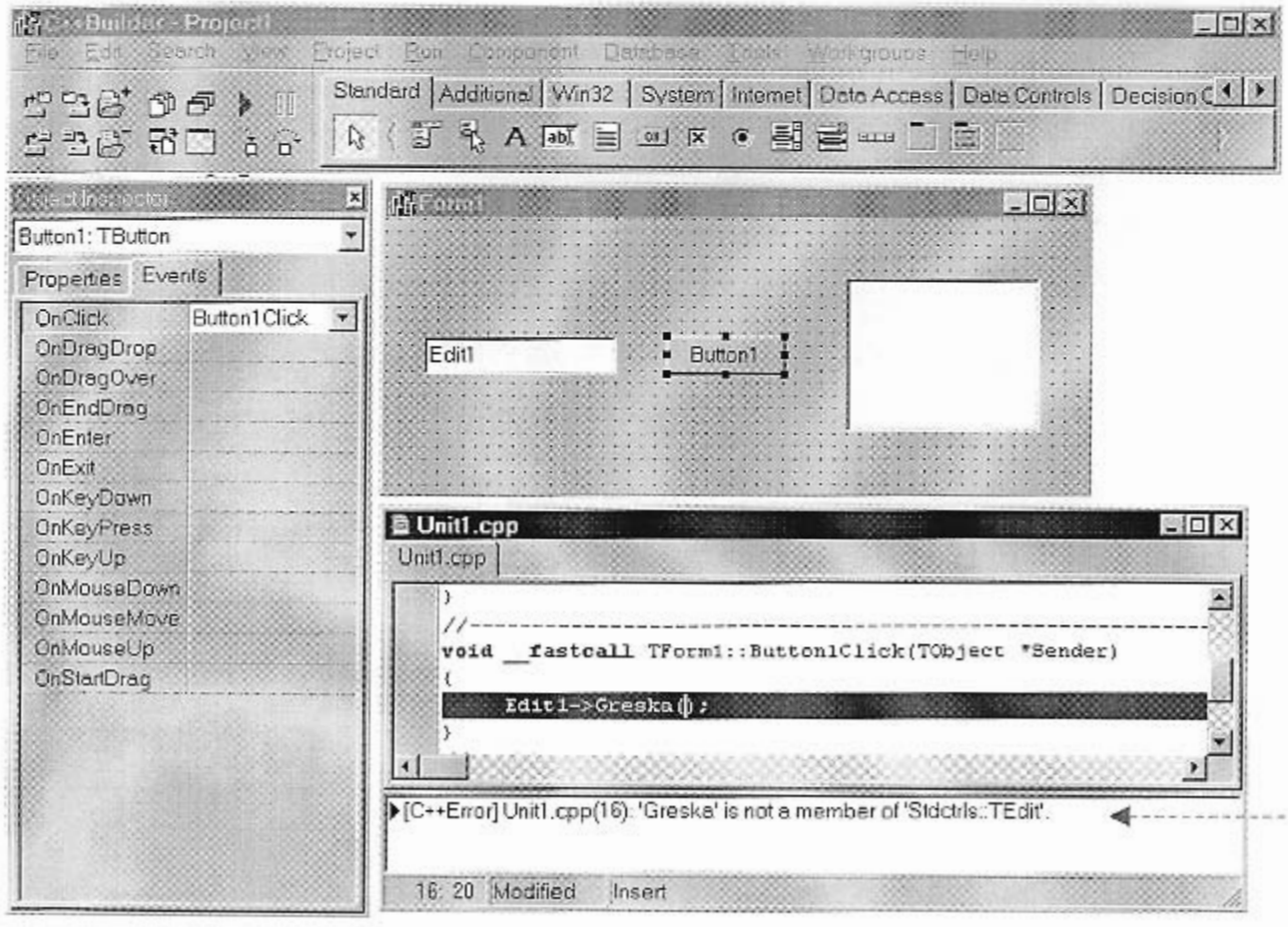
Startanjem projekta koji je do sada opisivan, dobija se sljedeća aplikacija:



2.5. Kreiranje izvršne aplikacije

Snimanjem projekta na disk i njegovim startanjem, na disku će se kreirati izvršna aplikacija (tipa EXE) u istom direktoriju u kojem je snimljen projekat i sa imenom koje je dato projektu.

2.6. Otkrivanje i ispravljanje grešaka

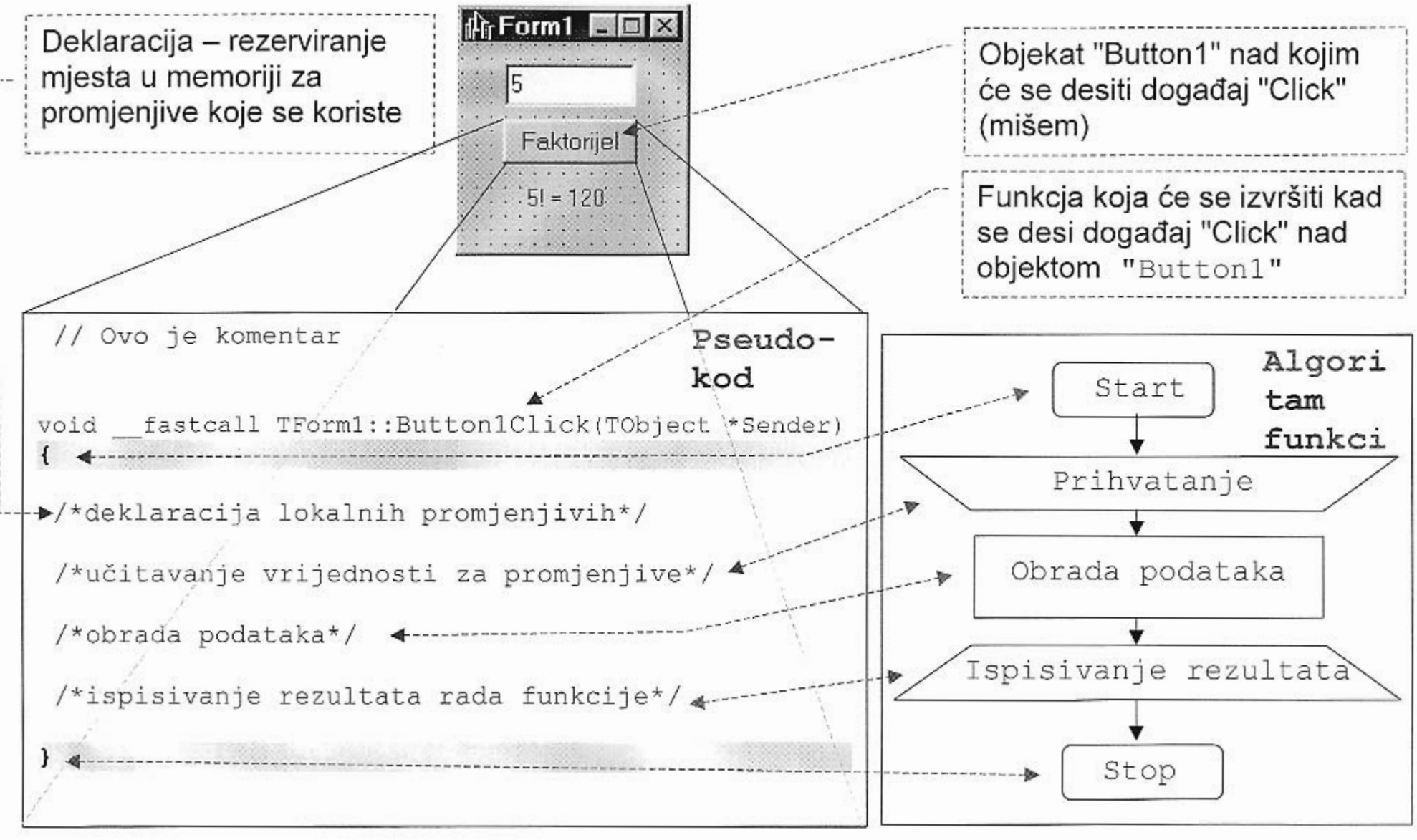


Spisak grešaka koje eventualno postoje u kodu aplikacije se dobija pri njenom startanju.

Ukoliko greške postoje, aplikacija neće biti startana, već će integrirano okruženje generirati listu grešaka na dnu prozora u kojem se nalazi kod aplikacije.

Pronalaženje greške u kodu se radi tako što se dvaput klikne na opis greške u listi.

8. Kodiranje događaja – tipična struktura funkcije čije je izvršavanje povezano sa događajem...



11. Jednostavna aplikacija

Praktičan primjer: Kreirati program koji računa sumu dva cijela broja.

Label

Panel

Deklaracija (rezerviranje mjesta u memoriji računala) za promjenjive koje će se koristiti

Događaj "Click" nad objektom "Button1" je iskodiran sljedećom funkcijom:

```
//---------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
 int x,y,z;

 x=StrToInt(Edit1->Text);
 y=StrToInt(Edit2->Text);

   z=x+y;

 Label3->Caption=IntToStr(z); // umjasto z moze x+y
}
//---------------------------------------------------------
```