Projektübung Informatik

Universelle Infrarot-Fernsteuerung

4. Objekt Orientiertes Design

Zweite, erweiterte Version

Gruppe 13: R. Zingg
M. Kaufmann
M. Stämpfli

Klasse: E4d

Datum: 28.9.1997

Inhaltsverzeichnis

4 Objektorientiertes Design

4.1 Allgemeines

4.1.1 Stand der Arbeit

Lauffähiges Programm: UIF Beta 2.0 (Prototyp 9) jedoch ohne Prototyp 7

D.h. es sind folgende Eigenschaften vorhanden:

- Das Hauptprogramm erscheint als Fenster (MDI) mit reduzierter Menüleiste.
- Umwandlung einer IR-Info in ein für ISES verständliches Format und Übertragung dieser Daten über die RS-232 an die Hardware.
- Eine Fernsteuerung mit den Elementen Button, Linie und Text kann von einer Datei geladen und benutzt werden.
- Die Hardware kann die Daten von der RS-232 empfangen, auswerten und eine Infrarotdiode ansteuern.
- Beliebig viele Fernsteuerungen mit beliebig vielen Elementen (Buttons, Linien und Texte) können kreiert, bearbeitet und gespeichert werden.

4.1.2 Entwicklungsumgebung

Zur implementation des Programms wird Borland C++ Builder Standard Version 1.0 Deutsch verwendet.

4.1.3 Schnittstellen

4.1.3.1 Fileaufbau der .irc Datei

4.1.4 Der Aufbau einer *.irc Datei:

Die *.irc Datei ist im CSV Format realisiert. D.h. es enthält einzelne Werte, getrennt durch Kommas.
Der erste Wert einer Zeile ist ein Schlüsselwort für die nachfolgenden Werte.
Folgende Schlüsselwörter können auftreten:

- RCForm: Beschreibt die Grösse und die Position einer Fernsteuerung.
- Text: Text zum Beschriften einer Fernsteuerung.
- Line: Linie zur Verbesserung der Übersichtlichkeit auf einer Fernsteuerung.
- Button: Taste einer Fernsteuerung.

Parameter zu RCForm:

1. Abstand der linken Seite der Fernsteuerung zum linken Rand des UIF Hauptfensters.

2. Abstand der oberen Seite der Fernsteuerung zum oberen Rand des UIF Hauptfensters.
3. Horizontale Grösse der Fernsteuerung.
4. Vertikale Grösse der Fernsteuerung.

Bsp.: *RCForm,0,0,480,270*

Parameter zu Text:

1. Abstand der linken Seite des Textes zum linken Rand der Fernsteuerung.
2. Abstand der oberen Seite des Textes zum oberen Rand der Fernsteuerung.
3. Anzuzeigender Text.

Bsp.: *Text,400,10,Testtext*

Parameter zu Line:

Bemerkung: Die Linie wird über die Diagonale einer Zeichenfläche dargestellt.

1. Abstand der linken Seite der Zeichenfläche zum linken Rand der Fernsteuerung.
2. Abstand der oberen Seite der Zeichenfläche zum oberen Rand der Fernsteuerung.
3. Breite der Zeichenfläche.
4. Höhe der Zeichenfläche.

Bsp.: *Line,400,35,53,0*

Parameter Button:

1. Abstand der linken Seite der Taste zum linken Rand der Fernsteuerung.
2. Abstand der oberen Seite der Taste zum oberen Rand der Fernsteuerung.
3. Breite der Taste.
4. Höhe der Taste.
5. Interner Name der Taste.
6. Text der auf der Taste erscheint (kann auch leer sein).
7. Pfad der Bitmap Datei die auf der Taste angezeigt werden soll (kann auch leer sein).
8. Typ der Fernsteuerung (S für Sony oder K für Kenwood).
9. Modulationsfrequenz in Hz.
10. Minimale Pulsbreite in µ Sekunden.
11. Anzahl Wiederholungen des zu sendenden Codes.
12. Code1 (für Sony und Kenwood).
13. Code2 (für Kenwood, muss bei Sony weggelassen werden).

Bsp.:

Button,10,10,100,25,SYSTEM_POWER,POWER,SYMB_POWER.BMP,S,40000,600,3,1111011
0101101011010101010101011000000000000000000000000000000000000000000000

4.1.4.1 Protokoll zu ISES

ISES funktioniert an COM 1 sowie an COM 2.
Die RS-232 Parameter sind folgendermassen definiert:

- Übertragungsrate: 9600 Bps
- Paritätsbit: none
- Datenbit: 8
- Stoppbit: 1

Das Übertragungsprotokoll sieht folgendermassen aus:

1.Byte:	Gibt an, ob ISES der IR-Code senden oder Empfangen soll.
2.Byte:	Gibt Auskunft, ob es sich um einen Sony oder Kenwood Code handelt.
3.Byte:	Gibt an, wieviel mal der Code ausgesendet werden soll.
4.Byte:	Bestimmt die Modulationsfrequenz des IR-Signals
5.& 6.Byte:	Bestimmt die Zeitdauer (in µSek.) eines Impulses des Nutz- oder Folgecodes.
Ab 7. Byte:	Wird der Nutzcode gesendet der nur ‘0’ und ‘1’ (in ASCII) enthalten darf. Nach dem Nutzcode Signalisiert das ‘E’ das Ende des Nutzcodes. Falls beim 2.Byte Kenwood gewählt wurde, kommt nun noch der Folgecode der wieder durch ein ‘E’ abgeschlossen wird.

Senden								
X	X	X	X	XX	1011....1011	E	1110...0101	E
S:Senden **E**:Empfangen **T**:Text ausgeben	**S**:Sony **K**:Kenwood	Anzahl Wiederholungen (#254 = 254 mal #255 = endlos)	Modulatinsfrequenz (#230 = 38 kHz #231= 40 kHz #232=42kHz ma	Zeitbasis (T) in us = #65536 - #XX	Nutzcode (Kenwood & Sony)	Ende des Nutzcodes	Folgecode (nur Kenwood)	Ende Folgecode

4.1.5 Hinweise

4.1.5.1 Namensgebung der Klassen

Da von der Entwicklungsumgebung von Borland C++ Builder in einigen Fällen ein T vor dem Klassennamen vorausgesetzt wird, haben wir uns entschlossen allen Klassennamen einheitlich ein T voranzustellen.

4.1.5.2 Das Schlüsselwort __published

Das Schlüsselwort __published stellt in Borland C++ Builder eine Erweiterung der Bereiche private, protected und public dar. Die Sichtbarkeitsregeln für published-Elemente sind die gleichen wie die für public-Elemente.
Auf Elemente im published Bereich wird, obwohl es so aussieht, nicht direkt zugegriffen, sondern über speziell deklarierte Zugriffsfunktionen.

Bsp. Deklarationen in einer Klasse, die später im __published Teil deklariert wird.

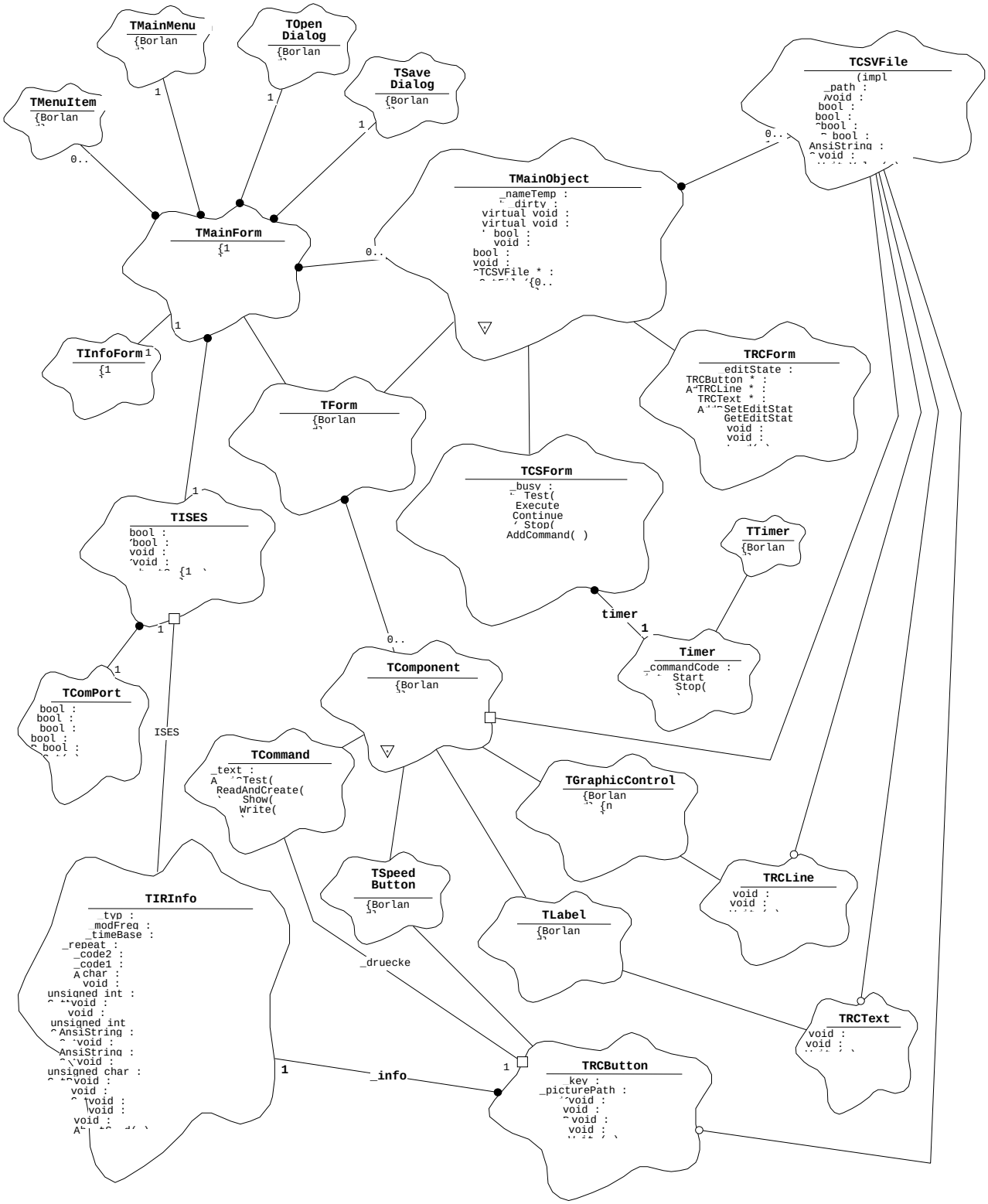
```
private:
  double FCustNo;
  double FOrderNo;
  double GetCustNo();
  double GetOrderNo();
  void SetCustNo(double NewCustNo);
  void SetOrderNo(double NewOrderNo);
public:           // User declarations
  __property double CustNo={read=GetCustNo,write=SetCustNo};
  __property double OrderNo={read=GetOrderNo,write=SetOrderNo};
```

4.2 Physikalisches Modell

Modul	Verantwortlich	Inhalt	Includes	Stand	Datum
TMainForm.h	R. Zingg	TMainForm	"TRCForm.h" "TISES.h" <vcl\ComCtrls.hpp> <vcl\ExtCtrls.hpp> <vcl\Messages.hpp> <vcl\Buttons.hpp> <vcl\Dialogs.hpp> <vcl\StdCtrls.hpp> <vcl\Menus.hpp> <vcl\Controls.hpp> <vcl\Forms.hpp> <vcl\Graphics.hpp> <vcl\Classes.hpp> <vcl\SysUtils.hpp> <vcl\Windows.hpp> <vcl\System.hpp> <vcl\ComCtrls.hpp>	fertig	28.9.97
TMainObject.h	R. Zingg	TMainObject	"TCSVFile.h" <dir.h> <vcl/forms.hpp>	fertig	28.9.97
TCSVFile.h	M. Stämpfli	TCSVFile	<string.h> <vcl/dstring.h> <windows.h>	fertig	28.9.97
TRCForm.h	R. Zingg	TRCForm	"TMainObject.h" "TCSVFile.h" "TRCButton.h" "TRCLine.h" "TRCText.h" <vcl\Controls.hpp> <vcl\Forms.hpp> <vcl\Graphics.hpp> <vcl\Classes.hpp> <vcl\Windows.hpp> <vcl\System.hpp> <vcl\Menus.hpp>	fertig	28.9.97
TRCButton.h	M. Kaufmann	TRCButton	"TIRInfo.h" "TCSVFile.h" <vcl/buttons.hpp>	fertig	28.9.97
TRCText.h	M. Kaufmann	TRCText	"TCSVFile.h" <vcl\Classes.hpp> <vcl\StdCtrls.hpp>	fertig	28.9.97
TRCLine.h	M. Kaufmann	TRCLine	"TCSVFile.h" <vcl\Classes.hpp> <vcl\Controls.hpp>	fertig	28.9.97
TIRInfo.h	M. Stämpfli	TIRInfo	"TISES.h" "TCSVFile.h"	fertig	28.9.97
TISES.h	M. Stämpfli	TISES	"TComPort.h" "TIRInfo.h" <string.h> <math.h> <vcl/dstring.h> <windows.h>	fertig	28.9.97
TComPort.h	M. Stämpfli	TComPort	<windows.h> <string.h> <vcl\dstring.h>	fertig	28.9.97
TSetupForm.h	R. Zingg	TSetupForm	"TISES.h" <vcl\ExtCtrls.hpp> <vcl\Controls.hpp> <vcl\Classes.hpp> <vcl\StdCtrls.hpp> <vcl\ExtCtrls.hpp> <vcl\Buttons.hpp> <vcl\StdCtrls.hpp> <vcl\Controls.hpp>	fertig	28.9.97

			<vcl\Forms.hpp> <vcl\Graphics.hpp> <vcl\Classes.hpp> <vcl\SysUtils.hpp> <vcl\Windows.hpp> <vcl\System.hpp>		
TButtonPropForm.h	R. Zingg	TButtonPropForm	"TRCButton.h" <vcl\Classes.hpp> <vcl\Controls.hpp> <vcl\StdCtrls.hpp> <vcl\Forms.hpp> <vcl\Mask.hpp> <vcl\ExtCtrls.hpp> <vcl\Buttons.hpp>	fertig	28.9.97
TInfoForm.h	R. Zingg	TInfoForm	<vcl\System.hpp> <vcl\Windows.hpp> <vcl\SysUtils.hpp> <vcl\Classes.hpp> <vcl\Graphics.hpp> <vcl\Forms.hpp> <vcl\Controls.hpp> <vcl\StdCtrls.hpp> <vcl\Buttons.hpp> <vcl\ExtCtrls.hpp>	fertig	1.7.97

4.3 Klassendiagramm



4.4 Klassenspezifikationen

4.4.1 Die Klasse TMainForm

4.4.1.1 Allgemeine Beschreibung

Das Hauptfenster von UIF ist ein Objekt der Klasse TMainForm. TMainForm beinhaltet das Hauptmenu und die Gadgetliste. TMainForm ist der Verwalter und Eigentümer aller MDI child Fenster. Child Fenster können Fernsteuerungen (TRemoteControl) oder Sequenzen (TSequence) sein. TMainForm beinhaltet ein Objekt der Klasse TISES, welches die Schnittstelle zu der Hardware ISES darstellt wird, sobald der User den Menubefehl "?|Info..." anklickt.

4.4.1.2 Headerfile der Klasse

```
#ifndef _TMAINFORM_H
#define _TMAINFORM_H
//============================================================================
// Projekt       : UIF
// Modul Titel   : TMainForm
// Datei Typ     : Header Datei
// Dateiname     : TMainForm.h
// Programmierer : Reto Zingg     Klasse : e4d     Gruppe : 13
// Datum         : 4.6.97 Zin
//============================================================================
// Letzte Änderung (Datum, Programmierer):
//  4. 6.97 Zin
// 11. 6.97 Zin
// 24. 6.97 Zin
// 28. 6.97 Zin
// 23. 8.97 Zin
// 24. 9.97 Zin
// 27. 9.97 Zin
// 28. 9.97 Zin
//============================================================================
// Beschreibung dieses Moduls:
// Das Hauptfenster von UIF ist ein Objekt der Klasse TMainForm.
// TMainForm beinhaltet das Hauptmenu und die Gadgetliste.
// TMainForm ist der Verwalter und Eigentümer aller MDI child Fenster.
// Child Fenster können Fernsteuerungen (TRemoteControl) oder
// Sequenzen (TSequence) sein.
// TMainForm beinhaltet ein Objekt der Klasse TISES, welches die Schnittstelle
// zu der Hardware ISES darstellt.
//============================================================================
class TMainForm;
//----------------------------------------------------------------------------
#include "TRCForm.h"
#include "TISES.h"
//----------------------------------------------------------------------------
#include <vcl\ComCtrls.hpp>
#include <vcl\ExtCtrls.hpp>
#include <vcl\Messages.hpp>
#include <vcl\Buttons.hpp>
#include <vcl\Dialogs.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\Menus.hpp>
#include <vcl\Controls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\Graphics.hpp>
#include <vcl\Classes.hpp>
#include <vcl\SysUtils.hpp>
#include <vcl\Windows.hpp>
#include <vcl\System.hpp>
```

```
//---------------------------------------------------------------------------
class TMainForm : public TForm
{
private:
  //---------------------------------------------------------------------------
  // HAS Assoziationen
  //---------------------------------------------------------------------------
  TISES _ISES; // Hardware Schnittstelle zur Hardware ISES

  //---------------------------------------------------------------------------
  // Attribute
  //---------------------------------------------------------------------------
  int _RCCount;  // Zähler für automatische Namensgebung von neuen Fernsteuerungen.

  //---------------------------------------------------------------------------
  // Private Methoden
  //---------------------------------------------------------------------------
  TRCForm * __fastcall CreateRC(const String name);
  // Erzeugt eine neue Fernsteuerung als MDI Child
  // name: Pfad der Datei in welcher die RC gespeichert werden soll
  //---------------------------------------------------------------------------

  void __fastcall ShowHint(TObject *Sender);
  // Ereignisfunktion wenn der Hinweis gezeigt werden soll.
  //---------------------------------------------------------------------------

  void __fastcall LoadSettings();
  // Laden der gespeicherten Einstellungen von UIF
  //---------------------------------------------------------------------------

  void __fastcall SaveSettings() const;
  // Speichert die Einstellungen von UIF in die WIN95 Registry
  //---------------------------------------------------------------------------

__published:
  //---------------------------------------------------------------------------
  // Hauptmenü und seine Einträge
  TMainMenu *MainMenu1;
  TMenuItem *FileNewRCItem;
  TMenuItem *FileNewSequenceItem;
  TMenuItem *FilePopup;
  TMenuItem *FileNewPopup;
  TMenuItem *FileOpenItem;
  TMenuItem *FileCloseItem;
  TMenuItem *FileSaveItem;
  TMenuItem *FileSaveAsItem;
  TMenuItem *FileSaveAllItem;
  TMenuItem *N3;
  TMenuItem *FileSetupItem;
  TMenuItem *FileExitItem;
  TMenuItem *EditPopup;
  TMenuItem *EditCutItem;
  TMenuItem *EditCopyItem;
  TMenuItem *EditPasteItem;
  TMenuItem *EditDeleteItem;
  TMenuItem *WindowPopup;
  TMenuItem *WindowCascadeItem;
  TMenuItem *WindowTileItem;
  TMenuItem *WindowArrangeItem;
  TMenuItem *WindowMinimizeItem;
  TMenuItem *HelpPopup;
  TMenuItem *HelpInfoItem;

  //---------------------------------------------------------------------------
  // Panel mit seinen Gadgets
  TPanel       *SpeedPanel;
  TSpeedButton *OpenBtn;
  TSpeedButton *SaveBtn;
  TSpeedButton *CutBtn;
  TSpeedButton *CopyBtn;
  TSpeedButton *PasteBtn;
  TSpeedButton *ExitBtn;
```

```
//---------------------------------------------------------------------------
// Statuszeile am unteren Ende des Fensters
TStatusBar   *StatusBar;

//---------------------------------------------------------------------------
// Fileselect Box für Öffnen und Speichern
TOpenDialog  *OpenDialog;
TSaveDialog  *SaveDialog;
//---------------------------------------------------------------------------

void __fastcall UpdateMenuItems(TObject *Sender);
// Aktualisiert das Erscheinen der Menueinträge. Wenn kein MDI Child vorhanden
// ist, kann zB. der Menueintrag 'Datei|Schliessen' nicht angewählt werden.
//---------------------------------------------------------------------------

void __fastcall FormCreate(TObject *Sender);
// Ereignisfunktion wenn das Hauptformular erzeugt wird.
// Nimmt Einstellungen vor beim erzeugen des Hauptformulars
//---------------------------------------------------------------------------

void __fastcall FormDestroy(TObject *Sender);
// Ereignisfunktion wenn das Hauptformular zerstört wird.
//---------------------------------------------------------------------------

void __fastcall FileNewRCClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Datei|Neu|Fernsteuerung' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall FileOpenItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Datei|Öffnen' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall FileCloseItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Datei|Schliessen' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall FileSaveItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Datei|Speichern' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall FileSaveAllItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Datei|Alles speichern' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall FileSaveAsItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Datei|Speichern als' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall FileSetupItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Datei|Setup ISES...' angewählt wird.
// Zeigt den Setupdialog an
//---------------------------------------------------------------------------

void __fastcall FileExitItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Datei|Beenden' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall EditCutItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Bearbeiten|Ausschneiden' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall EditCopyItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Bearbeiten|Kopieren' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall EditPasteItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Bearbeiten|Einfügen' angewählt wird.
//---------------------------------------------------------------------------

void __fastcall EditDeleteItemClick(TObject *Sender);
// Ereignisfunktion wenn der Menueintrag 'Bearbeiten|Löschen' angewählt wird.
//---------------------------------------------------------------------------
```

```
  void __fastcall WindowCascadeItemClick(TObject *Sender);
  // Ereignisfunktion wenn der Menueintrag 'Fenster|Kaskadieren' angewählt wird.
  //---------------------------------------------------------------------------

  void __fastcall WindowTileItemClick(TObject *Sender);
  // Ereignisfunktion wenn der Menueintrag 'Fenster|Teilen' angewählt wird.
  //---------------------------------------------------------------------------

  void __fastcall WindowArrangeItemClick(TObject *Sender);
  // Ereignisfunktion wenn der Menueintrag 'Fenster|Symbole anordnen' angewählt wird.
  //---------------------------------------------------------------------------

  void __fastcall WindowMinimizeItemClick(TObject *Sender);
  // Ereignisfunktion wenn der Menueintrag 'Fenster|Alle minimieren' angewählt wird.
  //---------------------------------------------------------------------------

  void __fastcall HelpInfoItemClick(TObject *Sender);
  // Zeigt den Info Dialog an mit den Copyright angaben.
  //---------------------------------------------------------------------------

public:
  //---------------------------------------------------------------------------
  // public Methoden
  //---------------------------------------------------------------------------
  virtual __fastcall TMainForm(TComponent *Owner);
  // Konstruktor. Erzeugt ein Objekt der Klasse TMainForm, ein Objekt der Klasse
  // TISES (Has-Assoziation) und Objekte der Klassen TMainMenu, TMenuItem,
  // TSaveDialog und TOpenDialog (ebenfalls Has-Assotiation)
  //---------------------------------------------------------------------------

  TISES* __fastcall GetISES();
  // Erfragen des Hardwaretreibers ISES
  //---------------------------------------------------------------------------
};

//---------------------------------------------------------------------------
extern TMainForm * MainForm;
//---------------------------------------------------------------------------
#endif   // _TMAINFORM_H
```

4.4.2 Die Klasse TMainObject

4.4.2.1 Allgemeine Beschreibung

TMainObject ist die abstrakte Oberklasse von TRCForm und TCSForm. TMainObject enthält ein Objekt der Klasse TCSVFile welches für die Verbindung des MainObjects mit einer CSV Datei auf einem Laufwerk zuständig ist.

4.4.2.2 Headerfile der Klasse

```
#ifndef _MAINOBJECT_H
#define _MAINOBJECT_H
//============================================================================
// Projekt       : UIF
// Modul Titel   : RemoteControl
// Datei Typ     : Header Datei
// Dateiname     : TMainObject.h
// Programmierer : Reto Zingg     Klasse : e4d     Gruppe : 13
// Datum         : 4.6.97 Zin
//============================================================================
// Letzte Änderung (Datum, Programmierer):
//  4. 6.97 Zin
// 11. 6.97 Zin
// 28. 6.97 Zin
// 14. 8.97 Zin
// 27. 9.97 Zin
// 28. 8.97 Zin
//============================================================================
// Beschreibung dieses Moduls:
// TMainObject ist die abstrakte Oberklasse von TRCForm und TCSForm.
// TMainObject enthält ein Objekt der Klasse TCSVFile welches für die
// Verbindung des MainObjects mit einer CSV Datei auf einem Laufwerk
// zuständig ist.
//============================================================================
class TMainObject;
//----------------------------------------------------------------------------
#include "TCSVFile.h"
//----------------------------------------------------------------------------
#include <dir.h>          // Header Datei fuer MAXFILE und MAXEXT ... Konstanten
#include <vcl/forms.hpp>  // Header Datei fuer TForm

//----------------------------------------------------------------------------
class TMainObject : public TForm
{
private:
  //--------------------------------------------------------------------------
  // Attribute
  bool _dirty;          // TRUE falls Änderungen seit erstelln oder speichern
  bool _filePathTemp;   // TRUE falls der gespeicherte Dateiname von UIF gegeben wurde
  TCSVFile _file;       // CSV-Dateiobjekt

public:
  //--------------------------------------------------------------------------
  // Methoden
  virtual __fastcall TMainObject(TComponent *Owner);
  // Konstruktor. Erzeugt ein Objekt der Klasse TMeinObject, zusätzlich ein
  // Objekt der Klasse TCSVFile (Has-Assotiation)
  //--------------------------------------------------------------------------

  bool __fastcall GetDirty() const;
  // Gibt den dirty Status zurück.
  // TRUE bedutet, dass seit dem erstellen oder letzten mal speichern
  //      eine Änderung vorgenommen wurde.
  //--------------------------------------------------------------------------
```

```
  void __fastcall SetDirty(bool dirty = TRUE);
  // Setzt den dirty Status.
  //---------------------------------------------------------------------------

  bool __fastcall GetFilePathTemp() const;
  // Gibt den Temporärstatus des Dateinamens zurück.
  // TRUE bedeutet, dass der Dateiname von UIF vorgegeben wurde. Somit wird
  //      der User bei der speicher Operation nach einem Dateinamen gefragt.
  //---------------------------------------------------------------------------

  void __fastcall SetFilePathTemp(bool filePathTemp);
  // Setzt den Temporärstatus des Dateinamens.
  //---------------------------------------------------------------------------

  TCSVFile* __fastcall GetFile();
  // Gibt einen Zeiger auf das _file zurück.
  //---------------------------------------------------------------------------

  virtual void __fastcall Save()=0;
  // Speichert das MainObject (Fernsteuerung oder Sequenz) in die durch _file
  // spezifizierten Datei.
  //---------------------------------------------------------------------------

  virtual void __fastcall Load()=0;
  // Ladet das MainObject (Fernsteuerung oder Sequenz) aus der durch _file
  // spezifizierten Datei.
  //---------------------------------------------------------------------------
};

#endif // _MAINOBJECT_H
```

4.4.3 Die Klasse TCSVFile

4.4.3.1 Allgemeine Beschreibung

Das Objekt CSVFile gehört zum Objekt MainObject. Es ist die Schnittstelle zum File. Die Klasse TCSVFile enthält Funktionen wie Path setzten, Datei öffnen, Datei lesen, Datei schreiben und Datei schliessen.

4.4.3.2 Headerfile der Klasse

```
#ifndef _TCSVFile_H
#define _TCSVFile_H
//============================================================================
// Projekt          : UIF
// Modul Titel      : TCSVFile
// Datei Typ        : Header Datei
// Dateiname        : TCSVFile.h
// Programmierer    : Marc Stämpfli    Klasse : e4d    Gruppe : 13
// Datum            : 25.6.97 Stä
//============================================================================
// Letzte Änderung (Datum, Programmierer):
// 25. 6.97 Stä
// 25. 6.97 Stä
// 31. 6.97 Stä
// 11. 7.97 Stä
// 20. 9.97 Stä
// 27. 9.97 Stä
//============================================================================
// Beschreibung dieses Moduls:
// Das Objekt CSVFile gehört zum Objekt MainObject
// Es ist die Schnittstelle zum File
// Die Klasse TCSVFile enthält Funktionen wie Path setzten, Datei öffnen,
// Datei lesen, Datei schreiben und Datei schliessen.
//============================================================================

class TCSVFile;
//----------------------------------------------------------------------------
#include <string.h>
#include <vcl/dstring.h>
#include <windows.h>
//----------------------------------------------------------------------------
class TCSVFile
{
  private:
    AnsiString      _path;           // Pathname für die Filezugriffe
    HANDLE          _hFile;          // FileHandle wird durch open Fkt. gesetzt.
    bool            _firstValue;     // 1.Element einer Zeile im File
    int __fastcall  ReadFromFile(AnsiString *value)const;
    int __fastcall  WriteToFile(const AnsiString *value);
//----------------------------------------------------------------------------
  public:
    TCSVFile();                   // Erzeugt ein Objekt der Klasse TCSVFile
                                  // mit seinen Attributen.
//----------------------------------------------------------------------------
    ~TCSVFile(){}
//----------------------------------------------------------------------------

  void __fastcall SetPath(const AnsiString &path);
//  Beschreibung:  Setzt den Path der für open Funktionen verwendet wird.
//----------------------------------------------------------------------------

  AnsiString __fastcall GetPath()const;
//  Beschreibung: Gibt den in der Funktion SetPath() angegebenen Path zurück.

//  Vorbedingung: Pfad muss mit SetPath() gesetzt worden sein.
//----------------------------------------------------------------------------
```

```
  bool __fastcall OpenToRead();
//  Beschreibung : Öffnet die Datei zum lesen.
//                 Das File ist im FILE_SHARE_READ Modus geöffnet.
//                 Rückgabewert true, wenn File geöffnet werden konnte.

//  Vorbedingung : Die Funktion Setpath() muss vorher aufgerufen worden sein.
//                 Es darf keine Datei geöffnet sein
//---------------------------------------------------------------------------

  bool __fastcall OpenToWrite();   //Wird erst in der Phase 2 Implementiert
//  Beschreibung : Öffnet eine Datei mit dem Namen, der in Setpath gesetzt wurde.
//                 Rückgabewert true, wenn File geöffnet werden konnte.

//  Vorbedingung : Die Funktion Setpath() muss vorher aufgerufen worden sein.
//                 Es darf keine Datei geöffnet sein.
//---------------------------------------------------------------------------

  int __fastcall ReadValue(AnsiString * value)const;
/*  Beschreibung :  Liest ab akt. pos im geöffneten File bis zum 1. Komma oder
                    CR+LF. Vor und nach der Zeichenfolge werden die Leerzeichen
                    abgeschnitten und schreibt den String in den von value
                    bezeigten String.

                    Rückgabewert:    0 = keine Fehler beim Einlesen
                                     1 = Fehler beim einlesen
                                     2 = Datei ist fertig

   Vorbedingung:  Die Datei muss mit OpenToRead geöffnet worden sein. */
//---------------------------------------------------------------------------

  int __fastcall ReadValue(unsigned int * value)const;
/*  Beschreibung:   Liest den nächsten Wert aus der geöffneten Datei und
                    wandelt ihn in einen unsigned int.

                    Rückgabewert:    0 = keine Fehler beim Einlesen
                                     1 = Fehler beim einlesen
                                     2 = Datei ist fertig

    Vorbedingung:   Die Datei muss mit OpenToRead geöffnet worden sein.     */
//---------------------------------------------------------------------------

  int __fastcall ReadValue(int * value)const;
/*  Beschreibung:   Liest den nächsten Wert aus der geöffneten Datei und wandelt
                    ihn in einen int.

                    Rückgabewert:    0 = keine Fehler beim Einlesen
                                     1 = Fehler beim einlesen
                                     2 = Datei ist fertig

    Vorbedingung:   Die Datei muss mit OpenToRead geöffnet worden sein.     */
//---------------------------------------------------------------------------

  int __fastcall ReadValue(char * value)const;
/*  Beschreibung:   Liest das nächste Wert aus der geöffneten Datei und wandelt
                    ihn in einen char.

                    Rückgabewert:    0 = keine Fehler beim Einlesen
                                     1 = Fehler beim einlesen
                                     2 = Datei ist fertig

    Vorbedingung:   Die Datei muss mit OpenToRead geöffnet worden sein.     */
//---------------------------------------------------------------------------

  int __fastcall ReadValue(unsigned char * value)const;
/*  Beschreibung:   Liest das nächste Wert aus der geöffneten Datei und wandelt
                    ihn in einen unsigned char.

                    Rückgabewert:    0 = keine Fehler beim Einlesen
                                     1 = Fehler beim einlesen
                                     2 = Datei ist fertig

    Vorbedingung:   Die Datei muss mit OpenToRead geöffnet worden sein.     */
//---------------------------------------------------------------------------
```

```
  bool __fastcall Close()const;
//  Beschreibung : Schliesst eine geöffnete Datei

//  Vorbedingung : Datei muss vorher mit einer open Funktion geöffnet worden
//                 sein (Filhandle vorhanden)

//  Nachbedingung: Rückgabewert true, wenn File geschlossen werden konnte.
//-------------------------------------------------------------------------

 int __fastcall TCSVFile::WriteValue(const AnsiString * value);
/*  Beschreibung:   Schreibt den String an die aktuelle Position der geöffneten
                    Datei.

                    Rückgabewert:    0 = keine Fehler beim Schreiben
                                     1 = Fehler beim Schreiben

    Vorbedingung:   Die Datei muss mit OpenToWrite geöffnet worden sein.     */
//-------------------------------------------------------------------------

 int __fastcall TCSVFile::WriteValue( const int * value);
/*  Beschreibung:   Schreibt den int an die aktuelle Position der geöffneten
                    Datei.

                    Rückgabewert:    0 = keine Fehler beim Schreiben
                                     1 = Fehler beim Schreiben

    Vorbedingung:   Die Datei muss mit OpenToWrite geöffnet worden sein.     */
//-------------------------------------------------------------------------

 int __fastcall TCSVFile::WriteValue( const unsigned int * value);
/*  Beschreibung:   Schreibt den int an die aktuelle Position der geöffneten
                    Datei.

                    Rückgabewert:    0 = keine Fehler beim Schreiben
                                     1 = Fehler beim Schreiben

    Vorbedingung:   Die Datei muss mit OpenToWrite geöffnet worden sein.     */
 //-------------------------------------------------------------------------

 int __fastcall TCSVFile::WriteValue( const char * value);
/*  Beschreibung:   Schreibt den int an die aktuelle Position der geöffneten
                    Datei.

                    Rückgabewert:    0 = keine Fehler beim Schreiben
                                     1 = Fehler beim Schreiben

    Vorbedingung:   Die Datei muss mit OpenToWrite geöffnet worden sein.     */
//-------------------------------------------------------------------------

 int __fastcall TCSVFile::WriteValue( const unsigned char * value);
/*  Beschreibung:   Schreibt den int an die aktuelle Position der geöffneten
                    Datei.

                    Rückgabewert:    0 = keine Fehler beim Schreiben
                                     1 = Fehler beim Schreiben

    Vorbedingung:   Die Datei muss mit OpenToWrite geöffnet worden sein.     */
//-------------------------------------------------------------------------

 bool __fastcall TCSVFile::WriteNextLine();
//  Beschreibung : Wird gebraucht, um im File auf eine nächste Zeile zu schreiben.
//                 Rückgabewert ist true, wenn die Funktion erfolgreich war.

//  Vorbedingung: File muss zum schreiben geöffnet worden sein
//-------------------------------------------------------------------------
 };
#endif  // _TCSVFile_H
```

4.4.4 Die Klasse TRCForm

4.4.4.1 Allgemeine Beschreibung

Ein Objekt der Klasse TRCForm (Remote Control Form) stellt auf dem Bildschirm eine Fernsteuerung dar. Diese besteht aus einem MDI Child Fenster im UIF Fenster. Es können beliebig viele TRCButton, TRCText und TRCLine Objekt auf ihr plaziert werden.

4.4.4.2 Headerfile der Klasse

```
#ifndef _TRCFORM_H
#define _TRCFORM_H
//============================================================================
// Projekt        : UIF
// Modul Titel    : TRemoteControl
// Datei Typ      : Header Datei
// Dateiname      : TRemoteControl.h
// Programmierer  : Reto Zingg     Klasse : e4d     Gruppe : 13
// Datum          : 4.6.97 Zin
//============================================================================
// Letzte Änderung (Datum, Programmierer):
//  4. 6.97 Zin
// 11. 6.97 Zin
// 14.08.97 Zin
//============================================================================
// Beschreibung dieses Moduls:
// Ein Objekt der Klasse TRCForm (Remote Control Form) stellt auf dem Bildschirm
// eine Fernsteuerung dar. Diese besteht aus einem MDI Child Fenster im UIF
// Fenster. Es können beliebig viele TRCButton, TRCText und TRCLine Objekt auf
// ihr plaziert werden.
//============================================================================
class TRCForm;
//----------------------------------------------------------------------------
#include "TMainObject.h"
#include "TCSVFile.h"
#include "TRCButton.h"
#include "TRCLine.h"
#include "TRCText.h"
//----------------------------------------------------------------------------
#include <vcl\Controls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\Graphics.hpp>
#include <vcl\Classes.hpp>
#include <vcl\Windows.hpp>
#include <vcl\System.hpp>
#include <vcl\Menus.hpp>
//----------------------------------------------------------------------------
#define RCEXT ".irc"           // Extension für Fernsteuerung

//----------------------------------------------------------------------------
class TRCForm : public TMainObject
// Representiert eine Fernsteuerung auf dem Bildschirm.
{
private:
  BOOL _useMode;                // true: Benützen Modus;  false: Bearbeiten Modus
  TComponent* _activeComponent; // Legt im Bearbeiten Modus fest, welche
                                // Komponente (Button, Linie, Text) angewählt ist.
```

```
__published:
  //---------------------------------------------------------------------------
  // Pulldown Menu welches automatisch zum Hauptfenster zugefügt werden
  TMainMenu *RCMenu;
  TMenuItem *RC;
  TMenuItem *RCNew;
  TMenuItem *RCNewButton;
  TMenuItem *RCNewText;
  TMenuItem *RCNewLine;
  TMenuItem *RCEdit;
  TMenuItem *RCUse;
  TMenuItem *N1;
  TMenuItem *RCProperties;

  void __fastcall FormClose(TObject *Sender, TCloseAction &Action);
  // Ereignisfunktion die aufgerufen wird, wenn das Formular (Fernsteuerung)
  // geschlossen wird.
  //---------------------------------------------------------------------------

  void __fastcall RCNewButtonClick(TObject *Sender);
  // Ereignisfunktion die aufgerufen wird, wenn der Benutzer den Menueintrag
  // 'Fernsteuerung|Neu>|Taste' anwählt.
  //---------------------------------------------------------------------------

  void __fastcall RCPropertiesClick(TObject *Sender);
  // Ereignisfunktion die aufgerufen wird, wenn der Benutzer den Menueintrag
  // 'Fernsteuerung|Eigenschaften' anwählt.
  //---------------------------------------------------------------------------

  void __fastcall RCEditClick(TObject *Sender);
  // Ereignisfunktion die aufgerufen wird, wenn der Benutzer den Menueintrag
  // 'Fernsteuerung|Bearbeiten' anwählt. Versetzt die aktuelle Fernsteuerung in
  // den bearbeiten - Modus.
  //---------------------------------------------------------------------------

  void __fastcall RCUseClick(TObject *Sender);
  // Ereignisfunktion die aufgerufen wird, wenn der Benutzer den Menueintrag
  // 'Fernsteuerung|Benützen' anwählt. Versetzt die Fernsteuerung in den
  // benützen - Modus.
  //---------------------------------------------------------------------------

public:
  __fastcall TRCForm(TComponent *Owner);
  // Konstruktor
  //---------------------------------------------------------------------------

  void __fastcall TRCForm::SetUseMode();
  // Versetzt die Fernsteuerung in den Benützen Modus.
  // Alle in RCForm enthaltenen Komponenten werden ebenfalls in den Benützen
  // Modus versetzt.
  //---------------------------------------------------------------------------

  void __fastcall TRCForm::SetEditMode();
  // Versetzt die Fernsteuerung in den Bearbeiten Modus.
  // Alle in RCForm enthaltenen Komponenten werden ebenfalls in den Bearbeiten
  // Modus versetzt.
  //---------------------------------------------------------------------------

  TComponent* __fastcall GetActiveComponent();
  // Gibt den Zeiger auf die aktive Komponente (_activeComponent) zurück.
  //---------------------------------------------------------------------------

  void __fastcall SetActiveComponent(TComponent* component);
  // Setzt den Zeiger _activeComponent auf den Wert component
  // Vorbed.: component muss Zeiger auf TRCButton, TRCLine oder TRCText sein,
  //          oder NULL
  //---------------------------------------------------------------------------

  TRCButton * __fastcall AddRCButton();
  // Fügt der Fernsteuerung eine Taste zu.
  //---------------------------------------------------------------------------
```

```
  TRCLine * __fastcall AddRCLine();
  // Fügt der Fernsteuerung ein Linie zu
  //---------------------------------------------------------------------------

  TRCText * __fastcall AddRCText();
  // Fügt der Fernsteuerung ein Text zu
  //---------------------------------------------------------------------------

  void __fastcall Save();
  // Speichert die Fernsteuerung in der im Objekt TCSVFile vermerkten Datei.
  //---------------------------------------------------------------------------

  void __fastcall Load();        // Laden
  // Ladet die Fernsteuerung aus der im Objekt TCSVFile vermerkten Datei.
  //---------------------------------------------------------------------------
};

#endif  // _TRCFORM_H
```

4.4.5 Die Klasse RCElement

Die gemeinsame Oberklasse RCElement der Klassen RCButton, RCText und RCLine wie sie in der OOA beschrieben wurde musste eliminiert werden, da Borland C++ bei VCL Klassen keine Doppelvererbung zulässt.

4.4.6 Die Klasse RCText

4.4.6.1 Allgemeine Beschreibung

Ein Object der Klasse RCText erscheint auf dem Bildschirm, auf der Fernsteuerung (SRC) als ein Text. Er dient zur Verbesserung der Uebersichtlichkeit der Fernsteuerung. Ein Object der Klasse RCText gehört immer zu einem Objekt der Klasse RCForm. Die Klasse RCText erbt von der abstrakten Klasse TLabel (Borland). Die Klasse RCElement wurde eliminiert da VCL-Klassen (Borland) keine Doppelvererbung zulassen.

4.4.6.2 Headerfile der Klasse

```
#ifndef TRCTextH
#define TRCTextH
//============================================================================
// Projekt        : UIF
// Modul Titel    : TRCText
// Datei Typ      : Header Datei
// Dateiname      : TRCText.h
// Programmierer  : Marcel Kaufmann    Klasse : e4d     Gruppe : 13
// Datum          : 25.6.97 Ka
//============================================================================
// Letzte Änderung (Datum, Programmierer):
//  1. 7. 97 Ka
// 16. 7. 97 Ka
// 12. 8. 97 Ka
// 15. 9. 97 Ka
// 27. 9. 97 Ka
//============================================================================
// Beschreibung dieses Moduls:
// Ein Object der Klasse RCText erscheint auf dem Bildschirm, auf der
// Fernsteuerung (SRC) als ein Text. Er dient zur Verbesserung der
// Uebersichtlichkeit der Fernsteuerung. Ein Object der Klasse RCText
// gehört immer zu einem Objekt der Klasse RCForm. Die Klasse
// RCText erbt von der abstrakten Klasse TLabel (Borland).
// Die Klasse RCElement wurde eliminiert da VCL-Klassen (Borland)
// keine Doppelvererbung zulassen.
//============================================================================
#include "TCSVFile.h"       // Using Beziehung
#include <vcl\Classes.hpp>  // Für TComponent
#include <vcl\StdCtrls.hpp> // TLabel
//----------------------------------------------------------------------------
class TRCText : public TLabel
// Ein TRText erbt von einem TLabel.

{
private:

public:
  __fastcall TRCText(Classes::TComponent* Owner);
  // Konstruktor TRCText: erzeugt ein neues, leeres Text Objekt
  // und fügt dieses der aktullen Fernsteuerung an.

  //--------------------------------------------------------------------------

  int __fastcall Read(const TCSVFile * _file);
  // Liest die Position und den gespeicherten Text aus der TCSVFile Datei ein
  // und weist sie dem TRCText Objekt zu.
  // Die Daten werden in folgender Reihenfolge eingelesen:
  // Left, Top, Text.
  // Vorbedingung: Uebergebenes CSVFile muss zum lesen geöffnet sein.
  //               (TCSVFile::OpenToRead() muss bereits ausgeführt worden sein.)
  // Rückgabewert: Return 0 falls alles in Ordnung,
  //                Return 1 falls ein Fehler aufgetreten ist.
  //--------------------------------------------------------------------------
```

```
  int __fastcall Write( TCSVFile * _file);
  // Schreibt die Position und den Text mittels TCSVFile in die im
  // TCSVFile vermerkte Datei.
  // Die Daten werden in folgender Reihenfolge in die Datei geschrieben:
  // Left, Top, Text.
  // Vorbedingung: Uebergebenes CSVFile muss zum schreiben geöffnet sein.
  //              (TCSVFile::OpenToWrite() muss bereits ausgeführt worden sein.)
  // Rückgabewert: Return 0 falls alles in Ordnung,
  //               Return 1 falls ein Fehler aufgetreten ist.
  //---------------------------------------------------------------------------

  void __fastcall SetUseMode();
  // Ereignisfunktion die den Text in den Ursprünglichen Zustand versetzt.
  // Neues Verhalten: Beim Anklicken wird die ursprüngliche Darstellung
  //                  und der Benützen Modus wieder hergestellt.
  //---------------------------------------------------------------------------

  void __fastcall SetEditMode();
  // Ereignisfunktion die den Text in den Bearbeiten Modus versetzt.
  // Neues Verhalten: Beim Anklicken ändert die Darstellung und der Button wird
  //                  bei seinem parent im _activeComponent eingetragen.
  //---------------------------------------------------------------------------

  void __fastcall SetActive(TObject *Sender);
  // Ereignisfunktion die den Text in den Aktiven Zustand versetzt.
  // Sie verändert die Darstellung des Buttons und trägt sich bei seinem parent
  // (TRCForm) im _activeComponent ein.
  //---------------------------------------------------------------------------


  void __fastcall SetNotActive();
  // Ereignisfunktion die den Text in den Normalzustand zurück versetzt.
  // Sie stellt die ursprüngliche Darstellung wieder her und setzt den
  //  _activeComponent seines parents auf Null.
  //---------------------------------------------------------------------------
};
#endif // End TRCText.h
```

4.4.7 Die Klasse RCLine

4.4.7.1 Allgemeine Beschreibung

Ein Object der Klasse RCLine erscheint auf dem Bildschirm, auf der Fernsteuerung (SRC) als eine Linie. Sie dient zur Verbesserung der Uebersichtlichkeit der Fernsteuerung. Ein Object der Klasse RCLine gehört immer zu einem Objekt der Klasse RCForm. Die Klasse RCLine erbt von der abstrakten Klasse TGraphicControl (Borland). Die Klasse RCElement wurde eliminiert da VCL-Klassen (Borland) keine Doppelvererbung zulassen.

4.4.7.2 Headerfile der Klasse

```
#ifndef _TRCLINE_H
#define _TRCLINE_H
//===========================================================================
// Projekt        : UIF
// Modul Titel    : TRCLine
// Datei Typ      : Header Datei
// Dateiname      : TRCLine.h
// Programmierer  : Marcel Kaufmann    Klasse : e4d     Gruppe : 13
// Datum          : 25.6.97 Ka
//===========================================================================
// Letzte Änderung (Datum, Programmierer):
//  1. 7. 97 Ka
// 16. 7. 97 Ka
// 12. 8. 97 Ka
// 15. 9. 97 Ka
// 27. 9. 97 Ka
//===========================================================================
// Beschreibung dieses Moduls:
// Ein Object der Klasse RCLine erscheint auf dem Bildschirm, auf der
// Fernsteuerung (SRC) als eine Linie. Sie dient zur Verbesserung der
// Uebersichtlichkeit der Fernsteuerung. Ein Object der Klasse RCLine
// gehört immer zu einem Objekt der Klasse RCForm. Die Klasse
// RCLine erbt von der abstrakten Klasse TGraphicControl (Borland).
// Die Klasse RCElement wurde eliminiert da VCL-Klassen (Borland)
// keine Doppelvererbung zulassen.
//===========================================================================
#include "TCSVFile.h"       // Using Beziehung
#include <vcl\Classes.hpp>  // Für TComponent
#include <vcl\Controls.hpp> // TGraphicControl

//---------------------------------------------------------------------------

class TRCLine : public TGraphicControl
// Ein TRCLine erbt von einem GraphicControl.
{
private:

public:
  __fastcall TRCLine(Classes::TComponent* Owner);
  // Konstruktor TRCLine: erzeugt ein neues Objekt Linie mit der Länge
  // null und fügt dieses der aktullen Fernsteuerung an.

  //-------------------------------------------------------------------------
  void __fastcall Paint(void);
  // Bestimmt die Zeichnungsfläche, auf welcher dann das Objekt
  // Linie gezeichnet werden kann.

  //-------------------------------------------------------------------------
  int __fastcall Read(const TCSVFile* _file);
  // Liest die Position, die Länge und die Höhe einer Linie ein und weist sie
  // dem Objekt zu. Die Linie wird über die Diagonale mit Länge und Höhe eines
  // Rechteckes gezeichnet. Die Daten werden in folgender Reihenfolge eingelesen:
  // Left, Top, With, Hight.
  // Vorbedingung: Uebergebenes CSVFile muss zum lesen geöffnet sein.
  //              (TCSVFile::OpenToRead() muss bereits ausgeführt worden sein.)
  // Rückgabewert: Return 0 falls alles in Ordnung,
  //               Return 1 falls ein Fehler aufgetreten ist.
```

```
  //---------------------------------------------------------------------------

  int __fastcall Write(TCSVFile* _file);
  // Schreibt die Position und Länge einer Linie mittels TCSVFile in die im
  // TCSVFile vermerkte Datei.
  // Die Linie wird über die Diagonale eines Rechteckes gezeichnet.
  // Die Daten werden in folgender Reihenfolge in die Datei geschrieben:
  // Left, Top, With,Hight, _key, Caption, _picturePath, Attribute des IRInfos.
  // Vorbedingung: Uebergebenes CSVFile muss zum schreiben geöffnet sein.
  //              (TCSVFile::OpenToWrite() muss bereits ausgeführt worden sein.)
  // Rückgabewert: Return 0 falls alles in Ordnung,
  //               Return 1 falls ein Fehler aufgetreten ist.
  //---------------------------------------------------------------------------

  void __fastcall SetUseMode();
  // Ereignisfunktion die die Linie in den Ursprünglichen Zustand versetzt.
  // Neues Verhalten: Beim Anklicken wird die ursprüngliche Darstellung
  //                  und der Benützen Modus wieder hergestellt.
  //---------------------------------------------------------------------------

  void __fastcall SetEditMode();
  // Ereignisfunktion die die Linie in den Bearbeiten Modus versetzt.
  // Neues Verhalten: Beim Anklicken ändert die Darstellung und der Button wird
  //                  bei seinem parent im _activeComponent eingetragen.
  //---------------------------------------------------------------------------

  void __fastcall SetActive(TObject *Sender);
  // Ereignisfunktion die die Linie in den Aktiven Zustand versetzt.
  // Sie verändert die Darstellung des Buttons und trägt sich bei seinem parent
  // (TRCForm) im _activeComponent ein.
  //---------------------------------------------------------------------------


  void __fastcall SetNotActive();
  // Ereignisfunktion die die Linie in den Normalzustand zurück versetzt.
  // Sie stellt die ursprüngliche Darstellung wieder her und setzt den
  //  _activeComponent seines parents auf Null.
  //---------------------------------------------------------------------------
};
#endif // End _TRCLINE_H
```

4.4.8 Die Klasse RCButton

4.4.8.1 Allgemeine Beschreibung

Ein Objekt der Klasse RCButton erscheint auf der Fernsteuerung (SRC) als Taste. Jeder RCButton beinhaltet ein Objekt der Klasse IRInfo welches alle Informationen beinhaltet die nötig sind, um einen IR-Code zu generieren. Wird der RCButton betätigt, wird ISES veranlasst, den entsprechenden IR-Code zu senden und die dem entsprechenden RCButton eigene Funktion beim Zielgerät auszulösen. Dazu wird das eigene Objekt IRInfo benötigt. Ein Objekt der Klasse RCButton gehört immer zu einem Objekt der Klasse RCForm. Die Klasse RCButton erbt von der abstrakten Klasse TSpeedButton (Borland). Die Klasse RCElement wurde eliminiert da VCL-Klassen (Borland) keine Doppelvererbung zulassen.

4.4.8.2 Headerfile der Klasse

```
#ifndef _TRCBUTTON_H
#define _TRCBUTTON_H
//============================================================================
// Projekt        : UIF
// Modul Titel    : TRCButton
// Datei Typ      : Header Datei
// Dateiname      : TRCButton.h
// Programmierer  : Marcel Kaufmann    Klasse : e4d     Gruppe : 13
// Datum          : 4.6.97 MKa
//============================================================================
// Letzte Änderung (Datum, Programmierer):
//  1. 7. 97 Ka
// 16. 7. 97 Ka
// 12. 8. 97 Ka
// 15. 9. 97 Ka
// 27. 9. 97 Ka
//============================================================================
// Beschreibung dieses Moduls:
//
// Ein Objekt der Klasse RCButton erscheint auf der Fernsteuerung (SRC) als
// Taste. Jeder RCButton beinhaltet ein Objekt der Klasse IRInfo welches alle
// Informationen beinhaltet die nötig sind, um einen IR-Code zu generieren.
// Wird der RCButton betätigt, wird ISES veranlasst, den entsprechenden IR-Code
// zu senden und die dem entsprechenden RCButton eigene Funktion beim Zielgerät
// auszulösen. Dazu wird das eigene Objekt IRInfo benötigt.
// Ein Objekt der Klasse RCButton gehört immer zu einem Objekt der Klasse RCForm.
// Die Klasse RCButton erbt von der abstrakten Klasse TSpeedButton (Borland).
// Die Klasse RCElement wurde eliminiert da VCL-Klassen (Borland) keine
// Doppelvererbung zulassen.
//============================================================================
#include "TIRInfo.h"
#include "TCSVFile.h"
#include <vcl/buttons.hpp>  // für TSpeedButton

class TRCButton :  public TSpeedButton // ein RCButton erbt von einem SpeedButton.
{
  private:
    TIRInfo _info;
    // HAS Assoziation
    AnsiString _key;
    // Gibt den internen Namen der Taste an. Dieser muss pro Fernsteuerung
    // einmalig vorhanden sein (max. 15 Zeichen, keine Leerzeichen). Dieser
    // Wert wird benötigt, um die Taste von einer Sequenz aus anzusprechen.
    AnsiString _picturePath;
    // Pfad einer Bitmap-Datei(.bmp), welche auf der Taste angezeigt
    // werden soll.
    BOOL _active;
    // Gibt an, ob dies der aktive Button ist.
```

```
    void __fastcall Paint(void);
    //  Ueberladen der Original Paintfunktion um RCButton selektiert
    //  darstellen zu können.

    __fastcall TRCButton( const TRCButton &Button);
    // Kopierkonstruktor verbieten.

  public:
     __fastcall TRCButton(Classes::TComponent* Owner);
    // Konstruktor TRCButton: erzeugt ein neues Objekt der Klasse TRCButton und
    // und ruft den Konstruktor eines IRInfos (HAS Assoziation) auf.
    // Das neue RCButton Objekt wird der Komponentenliste des Owners hinzugefügt.
    //---------------------------------------------------------------------------

    void __fastcall Press(System::TObject* Sender, TMouseButton Button,
                          Classes::TShiftState Shift, int X, int Y);
    // Der der Taste zugewiesene IR-Code wird von ISES ausgesendet.
    //---------------------------------------------------------------------------

    void __fastcall Release(System::TObject* Sender, TMouseButton Button,
                            Classes::TShiftState Shift, int X, int Y);
    // Sendet den Befehl an ISES,die Uebertragung abzubrechen.
    //---------------------------------------------------------------------------

    int __fastcall Read(const TCSVFile * _file);
    // Liest die gespeicherten Daten eines RCButtons ein und weist sie dem
    // Objekt zu.
    // Die Daten werden in folgender Reihenfolge eingelesen:
    // Left, Top, With, Hight, _key,buffer, _picturePath, Attribute des IRInfos.
    // Vorbedingung: Uebergebenes CSVFile muss zum lesen geöffnet sein.
    //               (TCSVFile::OpenToRead() muss bereits ausgeführt worden sein.)
    // Rückgabewert: Return 0 falls alles in Ordnung,
    //               Return 1 falls ein Fehler aufgetreten ist.
    //               Return 2 falls beim Laden der Grafik ein Fehler
    //                        aufgetreten ist.
    //---------------------------------------------------------------------------

    int __fastcall Write( TCSVFile * _file);
    // Schreibt die Daten eines RCButtons mittels TCSVFile in die im TCSVFile
    // vermerkte Datei.
    // Die Daten werden in folgender Reihenfolge in die Datei geschrieben:
    // Left, Top, With,Hight, _key, Caption, _picturePath, Attribute des IRInfos.
    // Vorbedingung: Uebergebenes CSVFile muss zum schreiben geöffnet sein.
    //               (TCSVFile::OpenToWrite() muss bereits ausgeführt worden sein.)
    // Rückgabewert: Return 0 falls alles in Ordnung,
    //               Return 1 falls ein Fehler aufgetreten ist.
    //---------------------------------------------------------------------------

    AnsiString __fastcall GetKey() const;
    // Gibt den internen Namen der Taste (_key) zurück.
    //---------------------------------------------------------------------------

    void __fastcall SetKey(AnsiString Key);
    // Setzt den internen Namen der Taste.
    // Vorbedingung : _key maximal 15 Zeichen, keine Leerzeichen.
    //---------------------------------------------------------------------------

    AnsiString __fastcall GetPicturePath() const;
    // Gibt den Bild-Pfad der Taste (_picturePath) zurück.
    //---------------------------------------------------------------------------

    int __fastcall SetPicture(AnsiString PicturePath);
    // Lädt das im _picturePath abgespeicherte Bild und zeigt es auf dem Button an.
    // Rückgabewert: Return 0: Kein Fehler aufgetreten.
    //               Return 1: Fehler beim öffnen der Datei aufgetreten.
    //               Return 2: Falscher Dateiname eingegeben.
    //               Return 3: Zu grosse Grafik-Datei.
    //---------------------------------------------------------------------------

    TIRInfo* __fastcall GetIRInfo();
    // Gibt die Adresse des zur Klasse gehörenden IRInfos zurück.
    //---------------------------------------------------------------------------
```

```
    void __fastcall SetUseMode();
    // Ereignisfunktion die den Button in den Ursprünglichen Zustand versetzt.
    // Neues Verhalten: Beim Anklicken wird die ursprüngliche Darstellung
    //                  und der Benützen Modus wieder hergestellt.
    //---------------------------------------------------------------------------

     void __fastcall SetEditMode();
    // Ereignisfunktion die den Button in den Bearbeiten Modus versetzt.
    // Neues Verhalten: Beim Anklicken ändert die Darstellung und der Button
    //                  wird bei seinem parent im _activeComponent eingetragen.
    //---------------------------------------------------------------------------

    void __fastcall SetActive(TObject *Sender);
    // Ereignisfunktion die den Button in den Aktiven Zustand versetzt.
    // Sie verändert die Darstellung des Buttons und trägt sich bei seinem parent
    // (TRCForm) im _activeComponent ein.
    //---------------------------------------------------------------------------

    void __fastcall SetNotActive();
    // Ereignisfunktion die den Button in den Normalzustand zurück versetzt.
    // Sie stellt die ursprüngliche Darstellung wieder her und setzt den
    //  _activeComponent seines parents auf Null.
    //---------------------------------------------------------------------------
};
#endif // _TRCBUTTON_H
```

4.4.9 Die Klasse TIRInfo

4.4.9.1 Allgemeine Beschreibung

Das Objekt der Klasse TIRInfo verwaltet sämtliche Daten, die für ISES notwendig sind. Zudem enthält die Klasse Funktionen, um die Attribute zu lesen, den IR-Code an ISES zu senden und eine Funktion, welche die IR-Uebertragung unterbricht.

4.4.9.2 Headerfile der Klasse

```
#ifndef _TIRINFO_H
#define _TIRINFO_H
//============================================================================
// Projekt         : UIF
// Modul Titel     : TIRInfo
// Datei Typ       : Header Datei
// Dateiname       : TIRInfo.h
// Programmierer   : Marc Stämpfli    Klasse : e4d     Gruppe : 13
// Datum           : 4.6.97 Stä
//============================================================================
// Letzte Änderung (Datum, Programmierer):
//  4. 6.97 Stä
// 11. 6.97 Stä
// 31. 6.97 Stä
// 11. 7.97 Stä
// 20. 9.97 Stä
// 27. 9.97 Stä
//============================================================================
// Beschreibung dieses Moduls:
// Das Objekt der Klasse TIRInfo verwaltet sämtliche Daten, die für ISES
// notwendig sind. Zudem enthält die Klasse Funktionen, um die Attribute zu
// lesen, den IR-Code an ISES zu senden und eine Funktion, welche die
// IR-Uebertragung unterbricht.
//============================================================================
class TIRInfo ;
#include "TISES.h"
#include "TCSVFile.h"
class TIRInfo
{
  private:
    char            _typ;                // Gibt an, ob es sich um einen
                                         // Sony oder Kenwood Typ handelt
    unsigned int    _modFreq;            // Modulationsfrequenz
    unsigned int    _timeBase;           // kürzeste Zeiteinheit des Signals
    AnsiString      _code1, _code2;      // IR-Code
    unsigned char   _repeat;             // Gibt an, wieviel mal der IR-Code
                                         // gesendet werden soll
    TISES          *_pISES;
//----------------------------------------------------------------------------

  public:
    __fastcall TIRInfo();              // Erzeugt ein Objekt der Klasse TIRInfo
                                       // mit seinen Attributen.
//----------------------------------------------------------------------------
    ~TIRInfo() {}
//----------------------------------------------------------------------------

   void __fastcall SetTyp(char typ);
// Beschreibung: Setzt den Typ (S für Sony oder K für Kenwood)

// Vorbedingung: Es darf nur S oder K übergeben werden.
//----------------------------------------------------------------------------

   char __fastcall GetTyp()const;
// Beschreibung: Gibt den Typ (S für Sony oder K für Kenwood) zurück.

// Vorbedingung: SetTyp() muss vorher einmal aufgerufen worden sein.
//----------------------------------------------------------------------------
```

```
   void __fastcall SetModFreq(unsigned int modFreq);
// Beschreibung: Setzt die Modulationsfrequenz (in Herz)

// Vorbedingung: Uebergabewert muss im Bereich von 4000 bis 42000 liegen.
//---------------------------------------------------------------------------

   unsigned int __fastcall GetModFreq()const;
// Beschreibung: Gibt die Modulationsfrequenz in Herz zurück
//               z.B. 38000 für 38 kHz

// Vorbedingung: SetModFreq() muss vorher einmal aufgerufen worden sein.
//---------------------------------------------------------------------------

   void __fastcall SetTimeBase(unsigned int timeBase);
// Beschreibung: Setzt die Timebase (in mikrosekunden)

// Vorbedingung: Uebergabewert muss im Bereich von 100 bis 65536 liegen.
//---------------------------------------------------------------------------

   unsigned int __fastcall GetTimeBase()const;
// Beschreibung: Gibt die Timebase (in mikrosekunden) zurück

// Vorbedingung: SetTimeBase() muss vorher einmal aufgerufen worden sein.
//---------------------------------------------------------------------------

   void __fastcall SetCode1(AnsiString code1);
// Beschreibung: Setzt den Code1 (Nutzcode)

// Vorbedingung: Code darf nur '0'-er und '1'-er enthalten und
//               maximal 1024 Zeichen lang sein.
//---------------------------------------------------------------------------

   AnsiString __fastcall GetCode1()const;
// Beschreibung: Gibt den Code1 zurück

// Vorbedingung: SetCode1() muss vorher einmal aufgerufen worden sein.
//---------------------------------------------------------------------------

   void  __fastcall SetCode2(AnsiString code2);
// Beschreibung: Setzt den Code2 (Folgecode)
//               Nur beim Kenwoodtyp erforderlich

// Vorbedingung: Code darf nur '0'-er und '1'-er enthalten und
//               maximal 1024 Zeichen lang sein.
//---------------------------------------------------------------------------

   AnsiString __fastcall GetCode2()const;
// Beschreibung: Gibt den Code2 (Folgecode) zurück
//               Nur beim Kenwoodtyp erforderlich

// Vorbedingung: SetCode1() muss vorher einmal aufgerufen worden sein.
//---------------------------------------------------------------------------

   void __fastcall SetRepeat(unsigned char repeat);
// Beschreibung: Setzt die Anzahl Wiederholungen wobei 255 endlos entspricht.

// Vorbedingung: Uebergabewert darf  max. 255
//---------------------------------------------------------------------------

   unsigned char __fastcall GetRepeat()const;
// Beschreibung: Gibt die Anzahl Wiederholungen zurück
//               max. 255 wobei 255 endlos entspricht

// Vorbedingung: SetRepeat() muss vorher einmal aufgerufen worden sein.
//---------------------------------------------------------------------------

   void __fastcall GetIRInfos()const;
// Beschreibung: Wird es in Phase 3 implementiert
//---------------------------------------------------------------------------
```

```
   int __fastcall Read(const TCSVFile *file);
// Beschreibung: Ladet Attribute von Datei
//               Rückgabewert: 0 wenn keine Fehler aufgetreten sind
//                             1 bei Fehlern
//---------------------------------------------------------------------------

   int __fastcall Write( TCSVFile *file);
// Beschreibung: Schreibt Attribute auf Datei
//               Rückgabewert: 0 wenn keine Fehler aufgetreten sind
//                             1 bei Fehlern
//---------------------------------------------------------------------------

   void __fastcall Send()const;
// Beschreibung: Veranlasst ISES, den IR-Code auszusenden
//---------------------------------------------------------------------------

   void __fastcall AbortSend()const;
// Beschreibung: Veranlasst ISES, den Abbruchcode an die Hardware zu senden
//---------------------------------------------------------------------------
};

#endif // _TIRINFO_H
```

4.4.10 Die Klasse TISES

4.4.10.1 Allgemeine Beschreibung

Die Klasse TISES stellt die Schnittstelle zur Hardware dar.(Eine Art Treiber)
Es gibt jeweils nur ein ISES pro Hardware. Diese Klasse enthält Funktionen, um einen IR-Code von einer Fernbedienung einzulesen. Weiter hat es Funktionen für des Senden und Empfangen eines IR-Codes.

4.4.10.2 Headerfile der Klasse

```
#ifndef _TISES_H
#define _TISES_H
//============================================================================
// Projekt          : UIF
// Modul Titel      : TISES
// Datei Typ        : Header Datei
// Dateiname        : TISES.h
// Programmierer    : Marc Stämpfli     Klasse : e4d     Gruppe : 13
// Datum            : 4.6.97 Stä
//============================================================================
// Letzte Änderung (Datum, Programmierer):
//  4. 6.97 Stä
// 11. 6.97 Stä
// 30. 6.97 Stä
// 11. 7.97 Stä
// 20. 9.97 Stä
// 27. 9.97 Stä
//============================================================================
// Beschreibung dieses Moduls:
// Die Klasse TISES stellt die Schnittstelle zur Hardware dar.(Eine Art Treiber)
// Es gibt jeweils nur ein ISES pro Hardware.
// Diese Klasse enthält Funktionen, um einen IR-Code von einer Fernbedienung
// einzulesen. Weiter hat es Funktionen für des Senden und Empfangen eines
// IR-Codes.
//============================================================================

class TISES ;
#include "TComPort.h"
#include "TIRInfo.h"
#include <string.h>
#include <math.h>
#include <vcl/dstring.h>
#include <windows.h>
//----------------------------------------------------------------------------

class TISES
{
  private:
    TComPort    _comDev;          // Settings
    AnsiString  _comStr;          // Port


  public:
    TISES();                          // Erzeugt ein Objekt der Klasse TISES
                                      // mit seinen Attributen.
//----------------------------------------------------------------------------
    ~TISES(){}
//----------------------------------------------------------------------------
```

```
  bool __fastcall ScanIRCode()const;       // Wird erst in Phase 3 Implementiert
/* Beschreibung:  Weist HW ISES an, den IR-Code der sendenden Fernsteuerung
                  einzuscannen und intern abzuspeichern.

                  Die Zeiten der Flanken des IR-Codes sind im HW ISES
                  gespeichert und können mit der Methode GetIRCode()
                  abgeholt werden.
                  Der Rückgabewert ist wahr, falls HW ISES
                  angesprochen werden konnte, sonst falsch. */
//---------------------------------------------------------------------------

  bool __fastcall GetIRCode(); //Wird erst in Phase 3 implementiert
/*  Beschreibung: Fordert ISES auf, den gespeicherten IR-Code zu senden .
                  Falls der Code Fehlerfrei eingelesen wurde, wird true
                  zurückgegeben, sonst false */
//---------------------------------------------------------------------------

  void __fastcall SendIRCode(const TIRInfo * info);
/*  Beschreibung: Wandelt die IR-Infos in einen für ISES verständlichen Code
                  um und sendet diesen an das Objekt ComPort welches den neuen
                  Code über die RS-232 an ISES schickt.        */
//---------------------------------------------------------------------------

  void __fastcall AbortSend();
/*  Beschreibung: Generiert einen für ISES verständlichen Code, damit ISES
                  die Infrarotübermittlung abbricht. Der neue Code wird
                  dann an das Objekt ComPort gesendet, welches den neuen
                  Code über die RS-232 an ISES schickt. */
//---------------------------------------------------------------------------

  void __fastcall SetPort(AnsiString port);
//  Beschreibung: Setzt den Port für ISES (z.B. COM1)
//---------------------------------------------------------------------------

  AnsiString __fastcall GetPort()const;
//  Beschreibung: Gibt den Port von ISES (z.B. COM1)
//---------------------------------------------------------------------------

  void __fastcall LoadSettings();
//  Beschreibung: Laden der gespeicherten Einstellungen von ISES
//---------------------------------------------------------------------------

  void __fastcall SaveSettings()const;
//  Beschreibung: Speichern der Einstellungen von ISES



};
#endif  // _TISES_H
```

4.4.11 Die Klasse TComPort

4.4.11.1 Allgemeine Beschreibung

Das Objekt der Klasse TComPort ist für die Zugriffe der seriellen Schnittstelle zuständig. Die Klasse enthält Funktionen wie COM setzten, COM öffen, vom COM lesen und schreiben und COM schliessen.

4.4.11.2 Headerfile der Klasse

```
#ifndef _TCOMPORT_H
#define _TCOMPORT_H
//============================================================================
// Projekt         : UIF
// Modul Titel     : TComPort
// Datei Typ       : Header Datei
// Dateiname       : TComPort.h
// Programmierer   : Marc Stämpfli     Klasse : e4d     Gruppe : 13
// Datum           : 4.6.97 Stä
//============================================================================
// Letzte Änderung (Datum, Programmierer):
//  4. 6.97 Stä
// 11. 6.97 Stä
// 31. 6.97 Stä
// 11. 7.97 Stä
// 20. 9.97 Stä
// 27. 9.97 Stä
//============================================================================
// Beschreibung dieses Moduls:
// Das Objekt der Klasse TComPort ist für die Zugriffe der seriellen
// Schnittstelle zuständig. Die Klasse enthält Funktionen wie COM setzten,
// COM öffen, vom COM lesen und schreiben und COM schliessen.
//============================================================================
#include <windows.h>
#include <string.h>
#include <vcl\dstring.h>              // für AnsiString
//----------------------------------------------------------------------------
class TComPort
//
{
  private:
    HANDLE       _hCom;               // Port Handle
    COMMCONFIG   _comPortConfig;      // Configuration Record
    AnsiString   _portName;           // Name des geöffneten Ports

  public:
    TComPort(){}                      // Erzeugt ein Objekt der Klasse TComPort
                                      // mit seinen Attributen.
//----------------------------------------------------------------------------
    ~TComPort(){}
//----------------------------------------------------------------------------

    bool __fastcall Set();
//  Beschreibung : Setzt die Eigenschaften des Portes .
//                 Rückgabewert true, wenn Einstellung korrekt waren.
//----------------------------------------------------------------------------

    bool __fastcall Open(AnsiString portName);
//  Beschreibung : Öffnet den angegebenen Port.
//                 Rückgabewert true, wenn Port geöffnet werden konnte.

//  Vorbedingung : Port muss mit Set() gesetzt worden sein.
//----------------------------------------------------------------------------
```

```
    bool __fastcall Close();
//  Beschreibung : Schliesst den in Set() angegebenen Port.
//                 Rückgabewert true, wenn Einstellung korrekt waren.

//  Vorbedingung : Port muss mit Open() geöffnet worden sein.
//---------------------------------------------------------------------------

    bool __fastcall SendCharImm(char data)const;
//  Beschreibung : Sendet ein Zeichen vor allen anderen auf den geöffneten Port
//                 Rückgabewert true, wenn Zeichen erfolgreich gesendet wurde.

//  Vorbedingung:  Port muss mit Open() geöffnet worden sein.
//---------------------------------------------------------------------------

    bool __fastcall Send(AnsiString data, unsigned int size)const;
//  Beschreibung : Sendet data auf den geöffneten Port
//                 Rückgabewert true, wenn Zeichen erfolgreich gesendet wurde.

//  Vorbedingung:  Port muss geöffnet sein.
//---------------------------------------------------------------------------

    bool __fastcall Recieve(AnsiString *data, unsigned int maxSize)const;
//  Beschreibung : Liest Daten vom geöffneten Port
//                 Rückgabewert true, wenn data empfangen werden konnte.

//                 Wird erst in Phase 3 implementiert.

//  Vorbedingung:  Port muss geöffnet sein.
//---------------------------------------------------------------------------
    };
#endif  // _TCOMPORT_H
```

4.4.12 Die Klasse TSetupForm

4.4.12.1 Allgemeine Beschreibung

Von der Klasse TSetupForm wird zu beginn des Programs genau ein Objekt erstellt. Es dient zur Einstellung von Angaben für das ISES Objekt. Auf dem Bildschirm erscheint TSetupForm als ein Eingabefenster wenn der Benutzer den Menupunkt 'Datei|Einstellungen' anwählt.

4.4.12.2 Headerfile der Klasse

```
#ifndef _TSETUPFORM_H
#define _TSETUPFORM_H
//============================================================================
// Projekt        : UIF
// Modul Titel    : TSetupForm
// Datei Typ      : Header Datei
// Dateiname      : TSetupForm
// Programmierer  : Reto Zingg     Klasse : e4d     Gruppe : 13
// Datum          : 15.8.97 Zin
//============================================================================
// Letzte Änderung (Datum, Programmierer):
// 24.09.97 Zin
// 27.09.97 Zin
//============================================================================
// Beschreibung dieses Moduls:
// Von der Klasse TSetupForm wird zu beginn des Programs genau ein Objekt
// erstellt. Es dient zur Einstellung von Angaben für das ISES Objekt. Auf dem
// Bildschirm erscheint TSetupForm als ein Eingabefenster wenn der Benutzer den
// Menupunkt 'Datei|Einstellungen' anwählt.
//============================================================================
class TSetupForm;
//----------------------------------------------------------------------------
#include "TISES.h"
//----------------------------------------------------------------------------
#include <vcl\ExtCtrls.hpp>
#include <vcl\Controls.hpp>
#include <vcl\Classes.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\ExtCtrls.hpp>
#include <vcl\Buttons.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\Controls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\Graphics.hpp>
#include <vcl\Classes.hpp>
#include <vcl\SysUtils.hpp>
#include <vcl\Windows.hpp>
#include <vcl\System.hpp>

//----------------------------------------------------------------------------
class TSetupForm : public TForm
{
private:

__published:
  //--------------------------------------------------------------------------
  // Graphische Elemente auf dem Formular
  TButton *OKBtn;
  TButton *CancelBtn;
  TBevel *Bevel1;
  TLabel *Label1;

  //--------------------------------------------------------------------------
  // Eingabeelemente auf dem Formular
  TComboBox *_ISESPort;
```

```
public:
  virtual __fastcall TSetupForm(TComponent* AOwner);

  void __fastcall DoSetup(TISES* ises);
  // Zeigt das Einstellungsformular an, wo der Benutzer Einstallungen vornehmen
  // kann. Bestätigt er mit OK, werden die Werte geschrieben, sonst verworfen.
  //---------------------------------------------------------------------------
};
//---------------------------------------------------------------------------
extern TSetupForm *SetupForm;
//---------------------------------------------------------------------------
#endif  // _TSETUPFORM_H
};
```

4.4.13 Die Klasse TButtonPropForm

4.4.13.1 Allgemeine Beschreibung

TButtonPropForm stellt auf dem Bildschirm ein Formular dar, in welchem die Eigenschaften eines RCButton verändert werden kann. Ein Objekt der Klasse TButtonPropForm wird zu beginn des Programs erzeugt, jedoch nicht angezeigt. Das Formular wird angezeigt, wenn sich die aktuelle Fernsteuerung im 'Bearbeiten-Modus', das aktuelle Objekt in der Fernsteuerung ein Button ist und wenn der Benutzer den Menueintrag 'Fernsteuerung|Eigenschaften' anwählt.

4.4.13.2 Headerfile der Klasse

```
#ifndef _TBUTTONPROPFORM_H
#define _TBUTTONPROPFORM_H
//============================================================================
// Projekt       : UIF
// Modul Titel   : TButtonPropForm
// Datei Typ     : Header Datei
// Dateiname     : TButtonPropForm.h
// Programmierer : Reto Zingg     Klasse : e4d     Gruppe : 13
// Datum         : 5.9.97 Zin
//============================================================================
// Letzte Änderung (Datum, Programmierer):
// 24. 9.97 Zin
// 27. 9.97 Zin
// 28. 9.97 Zin
//============================================================================
// Beschreibung dieses Moduls:
//
// TButtonPropForm stellt auf dem Bildschirm ein Formular dar, in welchem die
// Eigenschaften eines RCButton verändert werden kann.
// Ein Objekt der Klasse TButtonPropForm wird zu beginn des Programs erzeugt,
// jedoch nicht angezeigt. Das Formular wird angezeigt, wenn sich die aktuelle
// Fernsteuerung im 'Bearbeiten-Modus', das aktuelle Objekt in der Fernsteuerung
// ein Button ist und wenn der Benutzer den Menueintrag
// 'Fernsteuerung|Eigenschaften' anwählt.
//============================================================================
class TButtonPropForm;
//----------------------------------------------------------------------------
#include "TRCButton.h"
//----------------------------------------------------------------------------
#include <vcl\Classes.hpp>
#include <vcl\Controls.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\Mask.hpp>
#include <vcl\ExtCtrls.hpp>
#include <vcl\Buttons.hpp>

//----------------------------------------------------------------------------
class TButtonPropForm : public TForm
{
```

```
private:

__published:
  //---------------------------------------------------------------------------
  // Graphische Elemente des Formulars
  TLabel *Label1;
  TLabel *Label2;
  TLabel *Label3;
  TLabel *Label4;
  TLabel *Label5;
  TLabel *Label6;
  TLabel *Label7;
  TLabel *Label8;
  TLabel *Label9;
  TLabel *Label10;
  TLabel *Label11;
  TLabel *Label12;
  TLabel *Label13;
  TLabel *Label14;
  TLabel *Label15;
  TPanel *Panel1;

  //---------------------------------------------------------------------------
  // Eingabeelemente des Formulars
  TEdit *_left;
  TEdit *_top;
  TEdit *_height;
  TEdit *_width;
  TEdit *_key;
  TEdit *_caption;
  TEdit *_symbol;
  TRadioButton *_typSony;
  TRadioButton *_typKenwood;
  TEdit *_code1;
  TEdit *_code2;
  TEdit *_timeBase;
  TEdit *_repeat;
  TEdit *_modFreq;

  //---------------------------------------------------------------------------
  // Bedienelemente des Formulars
  TBitBtn *BitBtn1;
  TButton *BT_Chancel;
  TButton *BT_OK;

  void __fastcall BT_OKClick(TObject *Sender);
  // Ereignisfunktion, wenn der Benutzer den OK Button betätigt. Die Werte aus
  // dem Formular werden überprüft, falls alle in Ordnung sind werden sie dem
  // Button zugewiesen.
  // Falls Fehler aufgetreten sind, wird eine entsprechende Fehlermeldung am
  // Bildschirm angezeigt. Nach Bestätigung dieser Fehlermeldung kann der
  // Benutzer nochmals änderungen an den Einstellungen vornehmen, oder alle
  // Änderungen verwerfen.
  //---------------------------------------------------------------------------

public:
  __fastcall TButtonPropForm(TComponent* Owner);
  // Konstruktor, erzeugt ein Objekt der Klasse TButtonPropForm
  //---------------------------------------------------------------------------

  int __fastcall EditButton(TRCButton* button);
  // Die Werte des Objektes Button werden in das Editierformular übertragen und
  // das Fromular modal angezeigt.
  //---------------------------------------------------------------------------
};
//---------------------------------------------------------------------------
extern TButtonPropForm *ButtonPropForm;
//---------------------------------------------------------------------------
#endif  // _TBUTTONPROPFORM_H
};
```

4.4.14 Die Klasse TInfoForm

4.4.14.1 Allgemeine Beschreibung

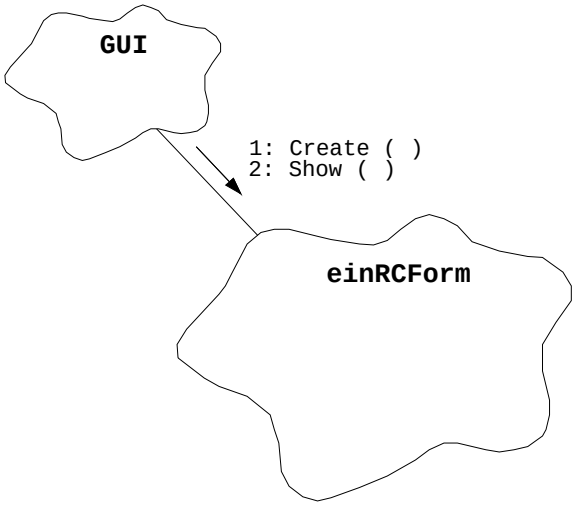
TInfoForm ist das Infofenster welches angezeigt wird, wenn auf den Menubefehl "?|Info..." geklickt wird. Ein Objekt der Klasse TInfoForm wird zum Programmbeginn erzeugt, aber nicht angezeigt. Wählt der User "?|Info..." an, so wird das Formular modal angezeigt.

4.4.14.2 Headerfile der Klasse

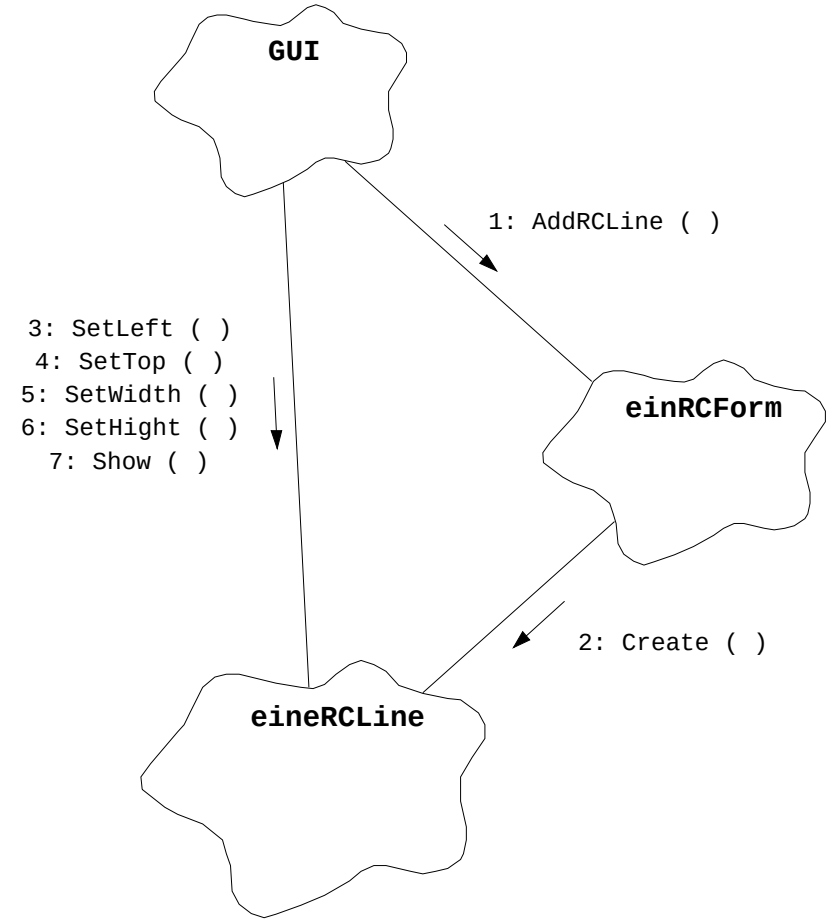
```
#ifndef _TINFOFORM_H
#define _TINFOFORM_H
//============================================================================
// Projekt       : UIF
// Modul Titel   : TInfoForm
// Datei Typ     : Header Datei
// Dateiname     : TInfoForm.h
// Programmierer : Reto Zingg     Klasse : e4d     Gruppe : 13
// Datum         : 25.6.97 Zin
//============================================================================
// Letzte Änderung (Datum, Programmierer):
// 1.7.97 Zin
//============================================================================
// Beschreibung dieses Moduls:
// TInfoForm ist das Infofenster welches angezeigt wird, wenn auf den Menubefehl
// "?|Info..." geklickt wird.
// Ein Objekt der Klasse TInfoForm wird zum Programmbeginn erzeugt, aber nicht
// angezeigt. Wählt der User "?|Info..." an, so wird das Formular modal angezeigt.
//============================================================================
class TInfoForm;
//----------------------------------------------------------------------------
#include <vcl\System.hpp>
#include <vcl\Windows.hpp>
#include <vcl\SysUtils.hpp>
#include <vcl\Classes.hpp>
#include <vcl\Graphics.hpp>
#include <vcl\Forms.hpp>
#include <vcl\Controls.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\Buttons.hpp>
#include <vcl\ExtCtrls.hpp>
//----------------------------------------------------------------------------
class TInfoForm : public TForm
{
private:

__published:
  //--------------------------------------------------------------------------
  // Graphische Elemente auf dem Formular
  TPanel *Panel1;
  TImage *ProgramIcon;
  TLabel *ProductName;
  TLabel *Version;
  TLabel *Copyright;
  TLabel *Comments;
  TLabel *Name1;
  TLabel *Name2;
  TLabel *Name3;
  TButton *OKButton;

public:
  virtual __fastcall TInfoForm(TComponent* AOwner);
  // Konstruktor. Erzeugt ein Objekt der Klasse TInfoForm mit seinen
  // Instanzvariablen.
  //--------------------------------------------------------------------------
};
extern TInfoForm *InfoForm;
//----------------------------------------------------------------------------
#endif  // _TINFOFORM_H
```

4.5 Objektszenarios

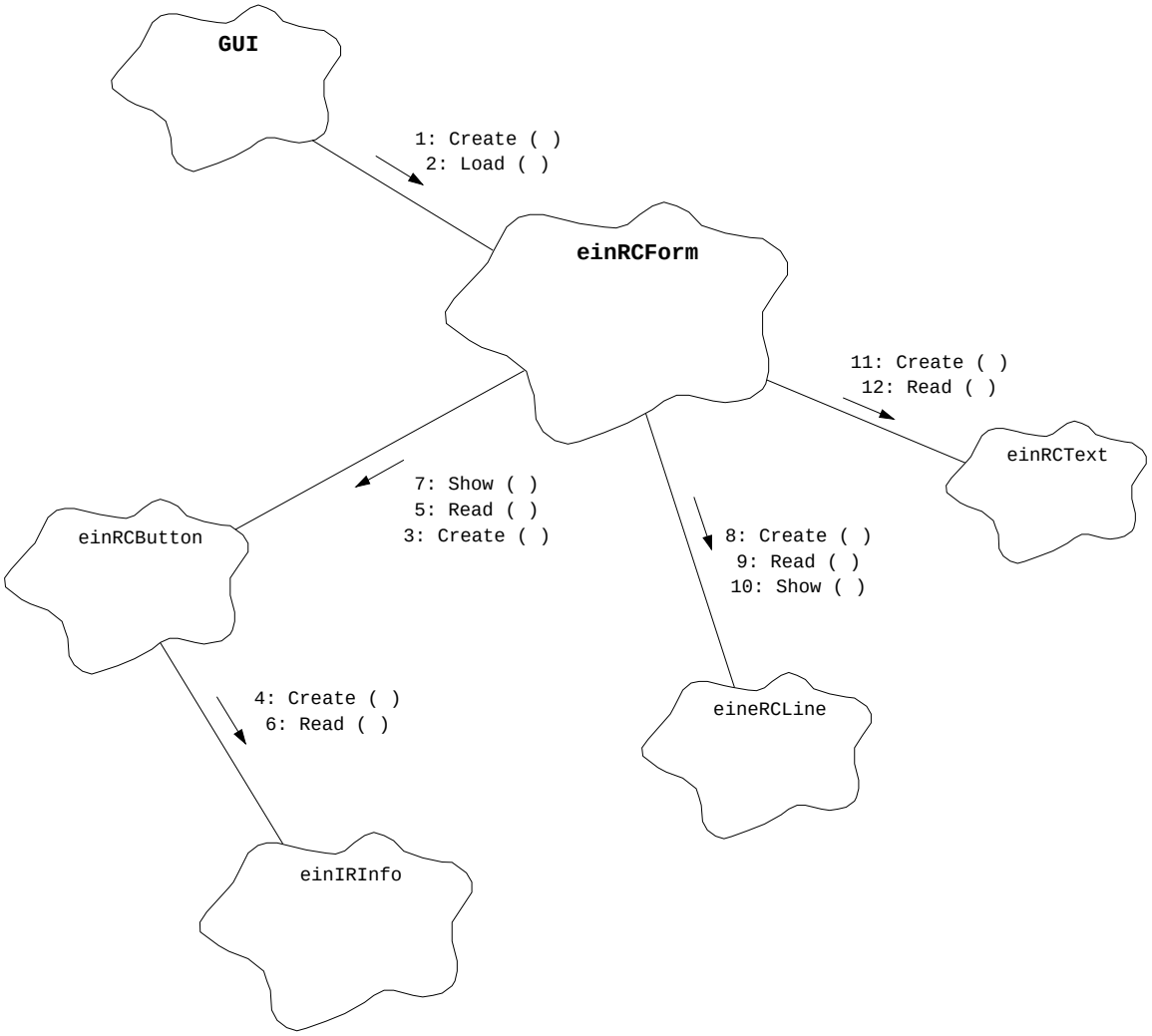
4.5.1 Objektszenario Fernsteuerung erstellen



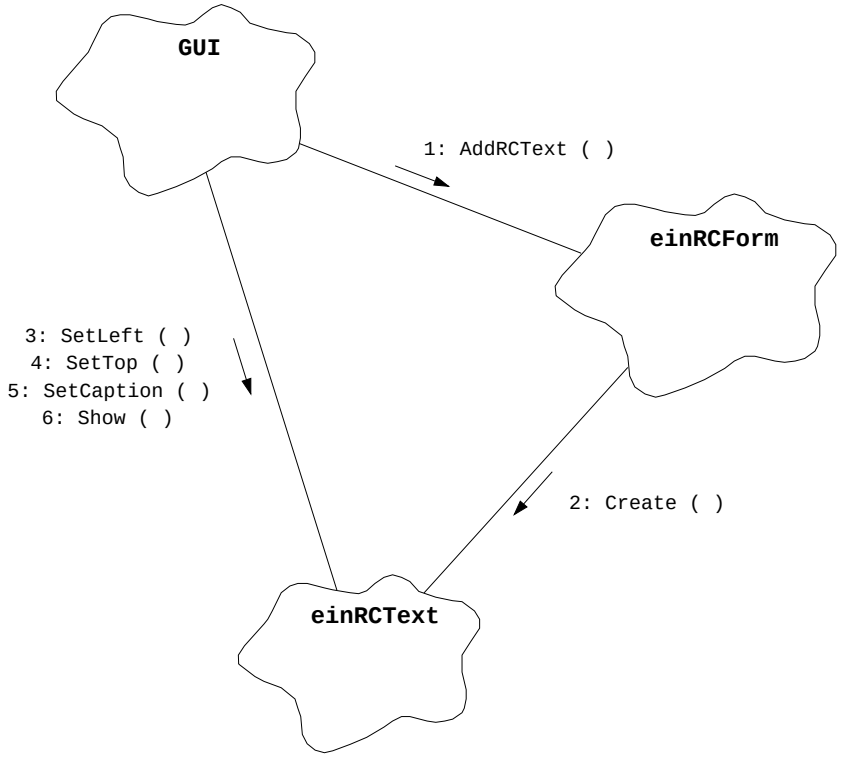
4.5.2 Objektszenario Linie erstellen



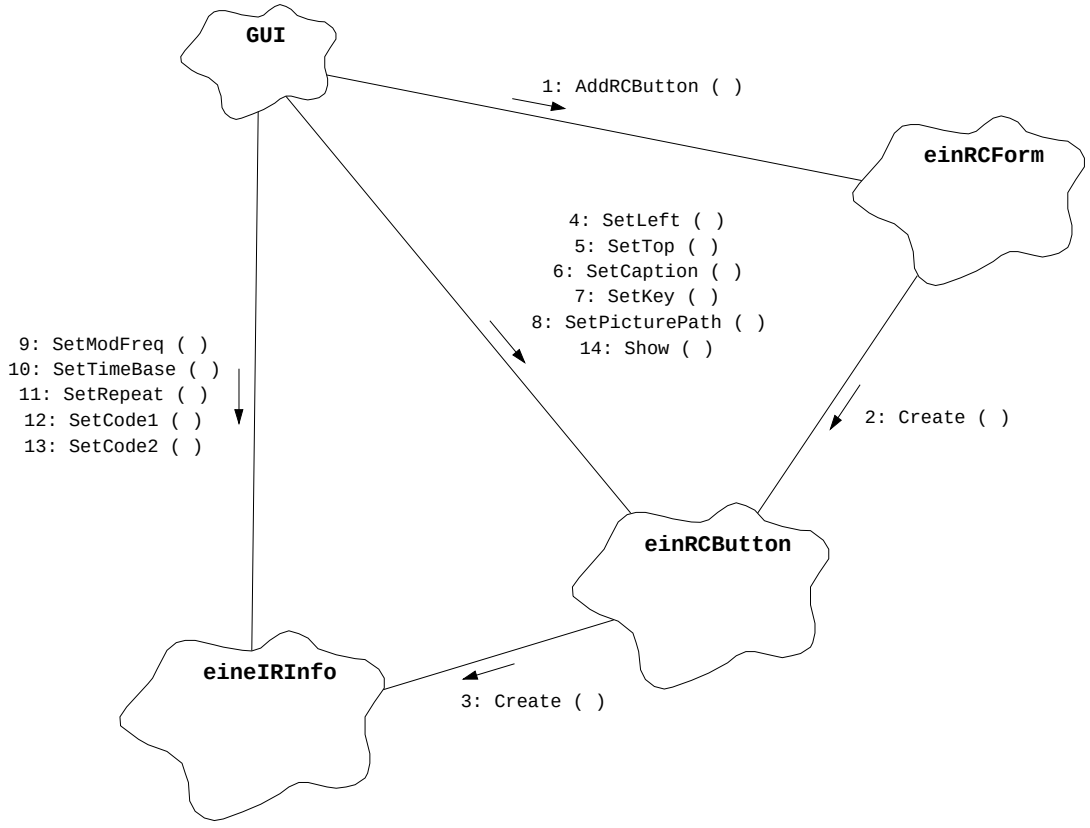
4.5.3 Objektszenario Fernsteuerung laden



4.5.4 Objektszenario Text erstellen



4.5.5 Objektszenario Button erstellen



4.5.6 Objektszenario IRCode senden