Technical Computing

Beschrijving van het opgeleverde product

Om de kaarten uit te lezen hebben we gebruik gemaakt van een TCP-server en Java framework. Van het framework hebben we een werkende smartcardreader gemaakt.

De TCP-server hebben we niet zelf gemaakt. Deze hebben we gekregen om de applicatie van de smartcardreader te testen. De reader is verbonden met de TCP-server. Deze verbinding komt tot stand door een TCP socket. Sockets worden gebruikt om te communiceren tussen applicaties via het netwerk of het internet.

De smartcardreader kan kaartnummers lezen via de TCP-server. Deze TCP-server kan ook een “dummycode” versturen. Hierbij wordt hetzelfde gedaan als wanneer er een smartcard wordt gescand, alleen hebben we dan geen echte smartcardreader nodig om de applicatie te testen. De kaartnummers van de smartcards zijn uniek en bestaan uit 8 bytes en een afsluit byte. Door middel van de “Refresh”-button kunnen de ingelezen kaartnummers die in de lijst staan worden verwijderd en staat er weer een lege lijst.

Aan het gegeven framework hebben we ook nog extra functies toegepast voor een hogere beoordeling. Een van de extra functies is het toevoegen van de datum en tijd waarop een kaartnummer is ingelezen. Een andere extra functie is de “Save”-button. De ingelezen kaartnummers kunnen nu worden opgeslagen.

Beschrijving van de toepassing van smartcards

Als de klant een keuze heeft gemaakt uit de collectie en zijn maten wil scannen, worden deze opgeslagen op een smartcard die de klant daarna krijgt. Als het een nieuwe klant is moet hij bij de counter zijn gegevens opgeven zodat hij geregistreerd wordt in het systeem. Als de klant de volgende keer weer in de winkel komt, hoeft hij niet meer in de bodyscan, tenzij de maten zijn veranderd. Hierdoor gaat het winkelproces sneller.

Hoe zijn we als team te werk gegaan?

Wij hebben besloten om de projectopdrachten te verdelen. We hebben deze opdracht dus niet met z’n allen gemaakt. Vidjai is aan deze opdracht begonnen en heeft steeds tussentijds zijn bevindingen met de groep gedeeld. Bij het maken van de opdracht zaten we wel met z’n allen bij elkaar aan het project. Ieder deed zijn deel van het project. Af en toen lieten we wat zien als we iets af hadden of als we iets niet snapten en gaven er uitleg bij.

Korte reflectie per teamlid

Vidjai Kalloe

Ik dacht eerst dat het een onmogelijke opdracht voor mij zou zijn. Omdat ik het toch interessant vond, heb ik voorgesteld om deze opdracht te gaan maken. Ik had wel hulp nodig van mijn teamgenoten en docenten, want anders zou ik er helemaal niet uitkomen. Het samenwerken binnen het projectgroepje ging erg goed.

Ik vond dat alles wel goed is gegaan, er hebben zich geen problemen opgedaan, behalve dat ik er soms niet uit kwam en dat ik dan hulp nodig had van anderen. Hierdoor hebben we het deels gezamenlijk gemaakt. Met deze opdracht heb ik meer geleerd over het programmeren in JAVA, maar alsnog heb ik er wel moeite mee. Ik vond het een leuke opdracht om te doen. Door deze opdracht te maken heb ik een beetje kunnen zien wat TC inhoud, want ik wilde misschien wel de richting TC kiezen. Mijn voorkeur gaat nog steeds uit naar SNE.

Maarten Lapére

TC is één van de richtingen die mij het meest aanspreken (samen met SE en GD), omdat ik erg van programmeren houd. Daarom heb ik er ook voor gekozen om Vidjai zo veel mogelijk te helpen bij deze opdracht. Het was leuk om hier samen aan te werken en ik heb een paar interessante nieuwe dingen geleerd over JAVA, bijvoorbeeld hoe je moet schrijven naar en lezen van een extern tekstbestand.

Django Drost

De colleges van Technical Computing waren in mijn optiek interessant. Het is waarschijnlijk de meest technische richting binnen de ICT en dat hebben we gemerkt. Het ontwikkelen van de applicatie was ondanks de onderbouwing en theorie moeilijk. Gelukkig hebben we door steeds samen de problemen te bespreken uiteindelijk een goed werkende applicatie kunnen maken. Bovendien heb ik geleerd hoe het nou eigenlijk achter de schermen werkt en eruit ziet bij het uitlezen van een SMART-card en dat zie je overal in de samenleving terug.

Joep van der Ben

Het was een mooie opdracht, waar je helaas niet met de hele groep aan had kunnen werken. Gelukkig heeft Vidjai het grootste deel in handen genomen, en met wat hulp van de rest zijn we tot een mooi resultaat gekomen. Voor mij zaten er wel een aantal leer momenten in zoals het verbinden met de TCP server en het opslaan van bestanden in een kladblok.

Jelle Wolkers

Het maken van het programma was sowieso lastig, dit kwam door een gebrek aan kennis. Er is vooral veel onderzoek vooraf gegaan aan het maken van het programma. Het contact maken met de server voor het uitlezen van de kaartnummers was het grootste struikelblok. Dit is uiteindelijk toch goed gekomen. Ik heb geleerd hoe je knoppen kunt linken aan functies ook al waren deze voor een deel al geconfigureerd. Daarnaast is het handig om te weten hoe je contact kunt maken met de server. De opdracht vond ik zeer leerzaam, en zal de kennis die ik heb opgedaan zeker utiliseren in de toekomst.

Raymond Wiering

Ik vond de opdracht van TC interessant om te doen. We leerden onder andere hoe we verbinding konden maken met de TCP server en hoe we een refreshknop konden maken. Aan het begin was het best lastig, maar na enig hulp van een docent waren we toch een eind op weg. Het laten uitlezen van de code was het moeilijkst om te ontwikkelen.

Code

```
/**
 * TC-project smartcards
 *
 * @author:  R. Slokker
 * @version  1 (28 mei 2007)
 *
 *
 */
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.net.*;
import java.io.*;
import java.util.*;
import java.text.*;

public class TCproject extends JFrame {

   // De klasse-attributen; de attributen die in main
   // gebruikt worden moeten static zijn.

   JTextArea txtInput;
   static JTextArea tekstarea;
   JButton startb;
   JButton refreshb;
   JButton saveb;
   static InputStream input;
   static boolean start;
   static Socket sockcl;
   static String str = new String("");
   static String date;

   public TCproject()
     {
          // Dit is de constructor van de klasse TCproject
          // Eerst wordt de constructor van de superklasse JFrame
          // uitgevoerd en daarbij wordt een frame voor de applicatie
          // aangemaakt; vervolgens wordt deze constructor uitgevoerd.

      setTitle("TCproject Smartcards");
      setSize(new Dimension(800, 600));
      setResizable(false);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

// Een panel is een onzichtbare component waarin user interface
// componenten kunnen worden gegroepeerd.
// Een pane is een bijzonder soort panel.
// Panels en panes kunnen vervolgens weer in een Frame worden
// geplaatst.
```

```
tekstarea=new JTextArea(20,20);
tekstarea.setEditable(false);
tekstarea.setForeground(Color.black);// kleur van de letters

JScrollPane panel1=new
JScrollPane(tekstarea,JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,JScrollPane.HORIZONTAL_SCROLLB
AR_ALWAYS);
panel1.setPreferredSize(new Dimension(200,800));
add(panel1,BorderLayout.WEST);
tekstarea.append("Ingelezen kaartnummers:\n\n");

// Plaats drie buttons in een panel en het panel
// vervolgens in het frame
      startb=new JButton("Start Reading");
      refreshb=new JButton("Refresh");
      saveb=new JButton("Save");
      JPanel panel2=new JPanel();
      panel2.add(startb);
      panel2.add(refreshb);
      panel2.add(saveb);
      add(panel2,BorderLayout.CENTER);

      ActionListener listener1=new StartHandler();
      startb.addActionListener(listener1);
      ActionListener listener2=new RefreshHandler();
      refreshb.addActionListener(listener2);
      ActionListener listener3=new SaveHandler();
      saveb.addActionListener(listener3);

      setVisible(true);
      start=false;

    }

  // Te ondernemen actie bij het klikken op de start-button
  class StartHandler implements ActionListener
  {
          public void actionPerformed(ActionEvent event)
          {
                  start=true;
          }
  }
```

```
// Te ondernemen actie bij het klikken op de refresh-button

  class RefreshHandler implements ActionListener
  {
          public void actionPerformed(ActionEvent event)
          {
                  // Plaats hier eigen code
                          tekstarea.setText("Ingelezen kaartnummers:\n\n"); //wist het scherm

          }
  }

  // Te ondernemen actie bij het klikken op de save-button
  class SaveHandler implements ActionListener
  {
                  public void actionPerformed(ActionEvent event)
                  {
                          FileWriter fWriter = null;
                          BufferedWriter writer = null;
                          try {
                                  fWriter = new FileWriter("Saved//SavedScans.txt");  // maakt of
                                  overschrijft een tekstbestand waar de gegevens in worden
                                  opgeslagen
                                  writer = new BufferedWriter(fWriter);

                                  writer.write("Opgeslagen gegevens met datum en tijd");  // Schrijft
                                  deze regel aan de top van het tekstbestand
                                  writer.newLine();   // Maakt een nieuwe regel aan
                                  writer.newLine(); // „          ,,          ,,          ,,          ,,
                                  writer.write(date + ":  " + str); // slaat de gegevens op in het
                                  tekstbestand

                                  writer.close();                 //sluiten van het schrijven naar bestand
                                  } catch (Exception e) {
                           }
                  }
  }

  //Functie om de huidige datum en tijd te verkrijgen
  public static String getCurrentTime() {
 //Maak een virtuele kalender aan
  Calendar cal = Calendar.getInstance();
 //Geef aan in welk formaat je de tijd wil hebben (Dag/Maand/Jaar - Uren:Minuten:Seconden
 SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy - HH:mm:ss");
 //Geef het antwoord op de aanvraag terug
 return sdf.format(cal.getTime());
}
```

```
public static void main(String[] args)
  {
     // Hier start het programma
      int i;

     TCproject tc=new TCproject(); // Constructor wordt uitgevoerd
     // Constructor van JFrame wordt eerst uitgevoerd

      // Wacht op het aanklikken van de start-button

     while(start==false){
                        try{
                                Thread.sleep(500);
                                }
                        catch(InterruptedException exception){};

                                        };

    // Maak hier verbinding met de TCP-server

          char[] data = new char[20];

         try{
                        sockcl = new Socket("127.0.0.1", 3000); // verbinding maken met de tcp server
                        InputStream input = sockcl.getInputStream(); //

                        int x;
                        char c;
                        while ( (i = input.read()) != -1) {

                        date = getCurrentTime();

                        for (i = 0; i < 8; i++) { // 8x herhalen om de gehele kaartnummer te weergeven
                        x = input.read(); // Lees één byte
                        data[i] = (char) x; //
                        str += data[i];    //

                        }

                        tekstarea.append(date+"\n");   //weergeeft de datum en tijdstip in het scherm
                        tekstarea.append(str+"\n");                 //weergeeft de kaartnummer in het scherm
                        str="";
                        }

                        }
                        catch (IOException ex)
                        {System.err.println(ex);}//        }
         }
}
```