ISSN: 1978 - 8282

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK PERSONAL ASSISTANT

Mohammad Irsan[1]
Denny Andwiyan[2]
Anil Ram[3]

Email: muhamad_irsan@yahoo.com; denny.a@osram.co.id; anil_cool@yahoo.com

Diterima: 8 Agustus 2009 / Disetujui: 24 Agusutus 2009

ABSTRACT

Along the time, the level of occupation will be a growing by an activity. This will cause the organizer needs to be something very important, to remember and record all activities that have to be implemented on time and planned. This recording can be done manually or through the electronic organizer. The problem is the organizer that there may have been reminded that an activity will be conducted in next day form of the alarm warning. To be able to solve the problem then designed a system that can record and remind about the activities that will be performed a warning alarm, Personal Digital Assistant (PDA) that is an organizer that can record and display a warning that comes with an alarm, is one application that can be is set to become one. This is done on the design of making the Personal Assistant, which is one function of the form of address book, password manager and task schedule. Results from this design is a system that could provide warning to the user in the form of an alarm event.

ABSTRAKSI

Seiring dengan berjalannya waktu, tingkat kesibukan akan aktivitas seseorang semakin bertambah. Hal ini menyebabkan kebutuhan akan organizer menjadi sesuatu yang sangat penting, untuk mengingat dan mencatat seluruh aktivitas yang ada agar dapat dilaksanakan tepat waktu dan terencana. Pencatatan ini dapat dilakukan melalui organizer manual maupun organizer elektronik. Permasalahannya adalah organizer yang ada saat ini belum dapat mengingatkan suatu aktivitas yang akan dilakukan dikemudian hari berupa alarm peringatan. Untuk dapat memecahkan masalah tersebut maka dirancang suatu sistem yang dapat mencatat serta mengingatkan mengenai aktivitas yang akan dilakukan berupa alarm peringatan, Personal Digital Assistant (PDA) yang merupakan sebuah organizer yang dapat mencatat serta menampilkan peringatan yang disertai dengan alarm, merupakan satu kesatuan aplikasi yang dapat diatur menjadi satu. Pada rancangan ini dilakukan pembuatan Personal

1. **Dosen Jurusan Teknik Informatika, STMIK Raharja**
   Jl. Jend Sudirman No.40 Modern Cikokol-Tangerang Telp 5529692
2. **Dosen Jurusan Teknik Informatika, STMIK Raharja**
   Jl. Jend Sudirman No.40 Modern Cikokol-Tangerang Telp 5529692
3. **Mahasiswa Jurusan Sistem Komputer, STMIK Raharja**
   Jl. Jend Sudirman No.40 Modern Cikokol-Tangerang Telp 5529692

Assistant yang merupakan satu kesatuan fungsi berupa address book, password manager dan task schedule. Hasil dari rancangan ini adalah suatu sistem yang dapat memberikan peringatan kepada user berupa alarm dalam melakukan suatu aktivitas.

Kata kunci : Agenda, PDA, Personal Assistant, Windows Pocket PC

PENDAHULUAN

Pada masa sekarang ini waktu merupakan hal yang sangat berharga terhadap seseorang yang sangat sibuk akan pekerjaanya. Dengan berjalannya waktu, tingkat kesibukan dari setiap orang semakin bertambah. Hal ini menyebabkan kebutuhan akan *organizer* menjadi sesuatu yang sangat penting, untuk mencatat dan mengingatkan mengenai seluruh aktivitas yang ada agar dapat dilaksanakan tepat waktu dan terencana. Pencatatan ini dapat dilakukan melalui *organizer* manual maupun *organizer* elektronik. *Organizer* elektronik mempunyai bentuk yang beraneka ragam dan yang paling digemari masyarakat umum adalah *organizer* yang dapat dibawa kemana saja atau biasa yang disebut dengan *mobile*.

Sekarang ini masyarakat sudah banyak dihadapkan dengan teknologi-teknologi yang dapat membantu mereka dalam menjalankan aktivitasnya. *Personal Digital Assistant* (PDA) merupakan salah satu *organizer* elektronik berbasis *mobile* yang sering dijumpai oleh masyarakat, yang dapat digunakan sebagai pengingat dalam melakukan suatu aktivitas. Dalam perangkat lunak PDA ini tersedia beberapa fungsi seperti *address book, task,* dan *alarm,* akan tetapi fungsi tersebut belum menjadi satu kesatuan aplikasi yang dapat diatur menjadi satu.

Untuk dapat melaksanakan suatu aktivitas secara terencana maka diperlukan suatu sistem yang dapat berinteraktif kepada user untuk dapat mencatat segala aktivitas yang berjalan kepada user. Suatu sistem yang tidak hanya dapat mencatat melainkan mengingatkan serta memberikan alarm peringatan ketika aktivitas yang dicatat harus dilaksanakan. Sehingga hal ini akan sangat membantu user dalam menyelesaikan pekerjaannya, terutama bagi orang-orang yang sangat sibuk akan kegiatannya.

PERUMUSAN PERMASALAHAN DAN RUANG LINGKUP

Bagaimana merancang suatu sistem *Personal Assistant* yang interaktif, serta menu *organizer* yang lengkap dan merupakan satu kesatuan. Ruang lingkup dari artikel ini berupa:

1. Menu *organizer*, yang berupa :

a. Task Schedule yang terdiri dari :

- *To-do* yaitu suatu aplikasi yang dapat mengatur aktivitas yang akan dilakukan.
- *Schedule* yaitu suatu aplikasi yang mengatur aktivitas yang akan dilakukan secara terus menerus dalam jangka waktu tertentu.
- *Events reminder* yaitu suatu aplikasi yang mengingatkan akan suatu *event* atau

kejadian. (dimana ketiga menu ini terintegrasi satu dengan yang lainnya)

b. Password manager yaitu suatu aplikasi untuk mengingatkan kita terhadap *username* dan *password* penting.

c. Address book yaitu suatu aplikasi yang digunakan untuk menyimpan data mengenai relasi.

2. Dijalankan pada PDA atau Pocket PC yang berbasis Windows Pocket PC

PEMBAHASAN

1. Personal Digital Assistant (PDA)

PDA merupakan salah satu *small device* berbasis *mobile* yang dapat dibawa dan digunakan dimana saja. Sebagian besar PDA dijalankan dengan sistem operasi Microsoft Windows Powered Pocket PC. Untuk membuat perangkat lunak didalam *small device*, dapat digunakan bahasa pemograman seperti Microsoft *Embedded* Visual Tools dan Microsoft Visual Studio.Net.

Tabel 1. Perbandingan kemampuan *device*

Device	CPU	Memory	Display
	450 MHz - 2.5 GHz	128 MB - 2 GB	1024x768 to 1600 x 1200 pixels 15 to 21 inch
	150 MHz - 296 MHz	16 - 64 MB	240x320 PIXELS, 6X8 cm 640 x 240 pixels, 16.5 cm
	Minimal	Minimal	5 lines of text 3 x 2.5 cm

2. Microsoft Visual Studio.Net Compact Framework

Microsoft.Net Compact Framework merupakan bagian dari Microsoft.Net Full Framework yang menyediakan fasilitas yang sama meskipun tidak secara keseluruhan. Fasilitas ini termasuk *common language runtime, just-in-time* (JIT) *compilation, evidencebased code security,* manajemen memori melalui *garbage collection, windows forms,* data

akses, XML, dan kemampuan untuk memakai XML *web services*.

Tabel 2. Batasan .Net Compact Framework

Fitur	Batasan pada NET Compact Framework
MDI Forms	Windows CE tidak mendukung Multiple Document Inferface
CDI+	Windows CE tidak mendukung GDI+
Drag-and-drop	Windows CE tidak mendukung drag-and drop
Printing	Windows CE tidak mendukung sistem printing
Web service	NET Compact Interface hanya mendukung client - side

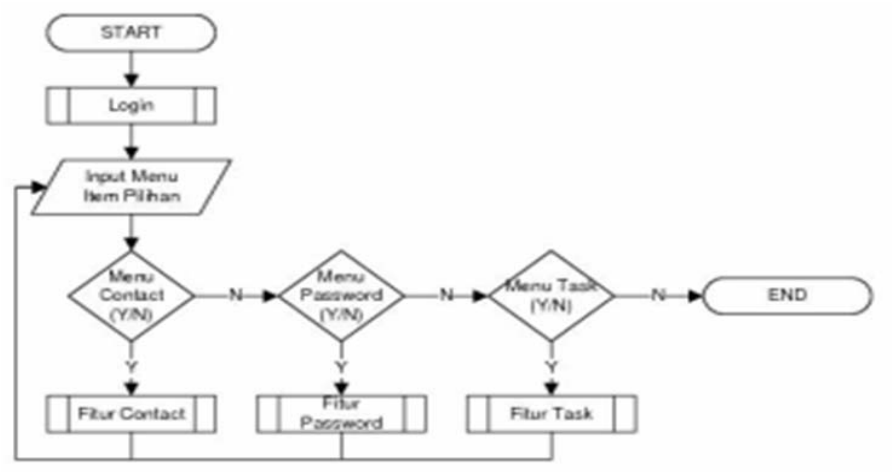
Dari table diatas dapat dilihat ada beberapa fungsi pada .Net Full Framework yang tidak disediakan oleh .Net Compact Framework. Yaitu :

- Menghubungkan dan mengirimkan *managed code dan* XML *web service* ke proses pengembangan aplikasi *smart device.*
- Mengoptimalkan sistem tenaga baterai dan mencegah pemakaian RAM dan CPU *cycle* dalam jumlah besar.
- Memastikan bahwa semua *managed resources* yang digunakan oleh aplikasi yang sedang berjalan, akan dibebaskan dan dikembalikan ke *host* sistem operasi jika aplikasi tersebut berhenti.
- Memastikan penggunaan RAM atau ROM yang disediakan oleh *device* agar aplikasi dapat tetap berjalan walaupun dalam keadaan *low of memory.*

Data yang dapat disimpan pada RAM atau ROM adalah kode asal yang menunjuk pada *common language runtime*, file yang berisi intruksi Microsoft Intermediate Language (MSIL) dan *metadata* untuk *class library.*

1. Desain Alur Proses Perangkat Lunak

Perancangan alur proses perangkat lunak ini bertujuan untuk memudahkan dalam pemahaman terhadap alur proses yang terjadi. Rancangan ini digambarkan dalam bentuk *flowchart*. Berikut gambaran alur proses *organizer*.



Gambar 1. *Flowchart Organizer*

Alur proses dimulai pada saat perangkat lunak dijalankan, dilanjutkan dengan proses *login*. Kemudian akan muncul tiga menu pilihan yaitu : menu *contact*, menu *password*, menu *task*. Jika salah satu menu tidak dipilih maka aplikasi akan terus menunggu *input*-an pilihan hingga program ditutup. Setiap menu yang dipilih, proses akan menuju fitur sesuai dengan menu yang terpilih.

2. Daftar Prosedur dalam Keseluruhan Perangkat Lunak

Untuk mempermudah dalam membaca prosedur-prosedur yang ada dibuat sebuah daftar prosedur beserta keterangannya. Berikut daftar prosedur yang dipakai dalam keseluruhan perangkat lunak :

Tabel 3. Daftar Prosedur

Fitur	Menu/Button	Procedure	Keterangan
Contact	Find a Name	textBox21_Text Changed	Mencari Nama
	Save	input_contact	Memasukkan Data
			Mencari posisi
		write_all_file_contact	Menulis ulang semua data
		write_file_contact	Memasukkan data (pertama kali saja)
		back	Kembali ke tabpage list
	View	listBox1_selected indexChanged	Menampilkan data ke tabpage view)

Fitur	Menu/Button	Procedure	Keterangan
	Edit	input_contact	Memasukkan data yang diubah
		search_pos_contact	Mencari posisi
		write_all_file_contact	Menulis ulang semua data
		back	kembali ke tabpage list
	Delete	write_all_file_contact	Menulis ulang semua data
		back	kembali ke tabpage list
	Delete All	button6_Click	Menghapus seluruh data
		back	kembali ke tabpage list
	Back	back	kembali ke tabpage list
Password	Find a Password	text9_TextChanged	Mencari Nama
	Save	input_password	Memasukkan data
		search_pos_password	Mencari posisi
		write_All_file_password	Menulis ulang semua data
		write_file_password	Memasukkan data (pertama kali saja)
		back	
	Save	listbox1_Selectedin dexChanged	Menampilkan data ke tabpage view
	Edit	input_password	Memasukkan data yang diubah
		search_pos_password	Mencari posisi
		write_All_file_password	Menulis ulang semua data
		back	Kembali ke tabpage list
	Delete	write_All_file_password	Menulis ulang semua data
		back	Kembali ke tabpage list
	Delete All	button1_click	Menghapus seluruh data
		back	Kembali ke tabpage list
	Back	back	Kembali ke tabpage list
Task	Save	input_task2	Memasukkan data
		union_new	

Fitur	Menu / Button	Procedure	Keterangan
		search_pos_task	Mencari posisi
		write_all_file_task	Menulis ulang semua data
		write_file_task	Memasukkan data (pertama kali saja)
		back	Kembali ke tabpage list
	View	listBox1_selectedIndexChanged	Menampilkan data ke tabpage view
	Edit	input_task3	Memasukkan data yang diubah
		iunion_new	Menggabungkan string subjek, tanggal, waktu, priority untuk mempermudah pengurutan data dari data yang diubah
		search_pos_task	Mencari posisi
		write_all_file_task	Menulis ulang semua data
		back	Kembali ke tabpage list
	Delete	write_all_file_task	Menulis ulang semua data
		back	Kembali ke tabpage list
	Delete periodic	button1_Click	Mendelete data antara selang waktu tertentu
		write_all_file_task	Menulis ulang semua data
		back	Kembali ke tabpage list
	Delete All	button6_Click	Menghapus seluruh data
		back	Kembali ke tabpage list
	Find	button7_Click	
	Test	PlaySound	Menyalakan suara dari alarm
	Save Path	button8_Click	Menyimpan path dari sound alarm
Task Check	Timer	tiner1_Tick	Mengecek apakah file cookis.dat ada atau tidak agar proses pengecekan dapat berjalan
		AlarmOn	Melakukan pengecekan kegiatan dengan alarm apa yang tanggal dan waktunya sama dengan tanggal dan waktu sekarang

Fitur	Menu/Button	Procedure	Keterangan
	don't Snooie	button1_Click	Menampilkan data ke tabpage view)

5. Implementasi Pada Fitur Contact

Deklarasi dan inisialisasi sistem dilakukan pertama kali saat program dijalankan. Deklarasi berupa jumlah daftar *contact* yang diperbolehkan, *path directory file* "cntct.dat" untuk menyimpan daftar *contact, array* utama untuk menyimpan data, variable *iname* yang merupakan *index listbox* ketika dipilih, variable j yang merupakan total daftar *contact* untuk *array*, dan *variable* I untuk proses *looping* global.

```
public struct Tcontact
        {  public string
        name,comp,email,haddr,htel,mob1tel,mob2tel,waddr,wtel,note;
        }
        //jumlah total daftar contact yang diperbolehkan
        private const int CONTACT_DATA = 500;
        //path directory file contact
        private const string FILE_CONTACT = "Program
        Files/SmartDeviceApplication/cntct.dat";
        //deklarasi tipe data global
        private Tcontact[] contact = new Tcontact[CONTACT_DATA];//array
contact
        private int iname,i,j;//iname =>index listbox yang dipilih, j =>total
        daftar contact, i => variabel looping global
        private int[] id = new int[CONTACT_DATA];
    private bool list_id = false;
```

Inisialisasi sistem dengan mengisi *array contact* dengan data yang ada dalam *file* "cntct.dat". sebelumnya dicek terlebih dahulu apakah *file* "cntct.dat" sudah ada atau belum jika belum maka proses inisialisasi pengisian tidak berjalan, sebaliknya jika *file* "cntct.dat" sudah ada maka proses inisialisasi pengisian berlangsung. Variable j diisi dengan jumlah data *contact* yang ada dalam *file* "cntct.dat" ditambah dengan nilai 1. Berikut fungsi-fungsi yang dibuat untuk mempermudah dalam penggunaan karena fungsi-fungsi tersebut dipanggil berulang kali :

```
//memasukan data ke array contact
private void input_contact(int n,Tcontact[] cntct,string nm,string
```

```
cm,string em,string ha,string ht,string mt1,string mt2,string wa,string
wt,string nt)
{
cntct[n].name=nm;cntct[n].comp=cm;cntct[n].email=em;cntct[n].haddr=ha;
cntct[n].htel=ht;
cntct[n].mob1tel=mt1;cntct[n].mob2tel=mt2;cntct[n].waddr=wa;cntct[n].w
tel=wt;cntct[n].note=nt;
 }
```

Fungsi *input-contact* digunakan untuk memasukkan data kedalam *array contact.*

```
//menulis data ke file contact
private void write_file_contact(int m,string fn,System.IO.FileMode
fm,System.IO.FileAccess fa)
{
FileStream fs = new FileStream(fn,fm,fa);
BinaryWriter bw = new BinaryWriter(fs);
bw.Write(contact[m].name);bw.Write(contact[m].comp);
bw.Write(contact[m].email);bw.Write(contact[m].haddr);
bw.Write(contact[m].htel);bw.Write(contact[m].mob1tel);
bw.Write(contact[m].mob2tel);bw.Write(contact[m].waddr);
bw.Write(contact[m].wtel);bw.Write(contact[m].note);
//kode pembatas alt+0133
bw.Write("...");
bw.Close();fs.Close();
  }
```

Fungsi write_file_*contact* digunakan untuk menulis 1 data dalam array ke file "cntct.dat", sedangkan untuk menulis semua data digunakan fungsi *write_all_file_contact.*

```
//menulis semua data dari array contact ke file contact
private void write_all_file_contact()
{ for (int w=0;w<=(CONTACT_DATA-1);w++)
{
if (contact[w].name != null)
{ if (w == 0)
```

```
{write_file_contact(w,FILE_CONTACT,FileMode.CreateNew,FileAcces
s.Write);}
else
{write_file_contact(w,FILE_CONTACT,FileMode.Append,FileAccess.W
rite);}
} } }
```

Untuk mengurutkan data yang masuk digunakan metode *insertion sort* dengan memanggil fungsi *search_pos_contact* . Pengurutan data dilakukan pada saat user menginputkan data. Jadi data akan teratur sesuai dengan urutan nama.

```
//mencari posisi index untuk memasukkan contact baru => insertion
sort
private void search_pos_contact(string nm,Tcontact[] cntct)
{
//nm adalah contact yang akan ditambahkan
int s=0;//index array contact diberi nilai 0
//mencari contact yang tidak null mencari posisi index untuk nm
//nilai CompareTo => a|a = 0 => a|b = -1 => b|a =1
while ((contact[s].name != null) && (nm.CompareTo(contact[s].name)
==
1))
{s++;}
//menggeser ke kanan 1 index untuk semua array, setelah index nm
ditemukan
for (int p=j-1;p>=s;p—)
{
contact[p+1]=contact[p];
}
//menulis contact ke posisi index nm
contact[s]=cntct[0];
}
```

Fungsi *view_list_contact* digunakan untuk menampilkan data *array contact* ke dalam tampilan *listbox* yang memperlihatkan nama dan nomor handphone saja.

```
//menampilkan daftar contact pada listbox
public void view_list_contact()
```

```
{
listBox1.Items.Clear();
tabPage1.Enabled=true;tabPage2.Enabled=true;tabPage3.Enabled=false;
if (File.Exists(FILE_CONTACT))
{
read_file_contact();
for (int v=0;v<=(CONTACT_DATA-1);v++)
{
if (contact[v].name != null)
{
if (contact[v].name.Length > 15)
{
listBox1.Items.Add(contact[v].name.Remove(15,contact[v].
name.Length-15)+" | "+contact[v].mob1tel+" |
"+contact[v].htel);
}
else
{
string cn="";
for (int icn=1;icn<=(15-contact[v].name.Length);icn++)
{cn=cn+" ";}
listBox1.Items.Add(contact[v].name+cn+" |
"+contact[v].mob1tel+" | "+contact[v].htel);
} } }
if (j == 1)
{contact_status.Text=j.ToString()+" contact";}
else
{contact_status.Text=j.ToString()+" contacts";}
button6.Enabled=true;
}
else
{
contact_status.Text="0 contact";
button6.Enabled=false;
} }
```

Fungsi-fungsi yang telah dibuat diatas digunakan dalam proses-proses yang terjadi dalam aplikasi contact, yaitu:

a. Proses *add new contact*

Proses untuk menambah jumlah contact. Dimulai dari pengecekan apakah nama ada isinya atau belum karena nama adalah primary key disini, jadi tidak diperbolehkan kosong. Jika kembar masih diperbolehkan. Kemudian dilakukan pengecekan ada tidaknya file "cntct.dat", pengecekan ini mempengaruhi fungsi apa yang akan dipanggil oleh proses. Dilanjutkan dengan pengurutan data dan menulis data array ke file "cntct.dat".

b. Proses *view* data

Proses ini untuk menampilkan data secara lengkap ke tabpage view sesuai dengan index listbox yang dipilih. Dalam tabpage ini terdapat 3 button, yaitu button back edit , dan delete.

c. Proses *edit* dan *delete contact*

Proses untuk melakukan perubahan terhadap data yang ada. Baik hanya diubah atau dihapus dari daftar

2. Implementasi Pada Fitur *Password*

Deklarasi dan inisialisasi sistem dilakukan pertama kali saat program dijalankan. Deklarasi berupa jumlah daftar *password* yang diperbolehkan, path directory *file* "psswrd.dat" untuk menyimpan daftar *password, array* utama untuk menyimpan data, variable i desc yang merupakan index listbox ketika dipilih, variabel j yang merupakan total daftar untuk *array*, dan variable i untuk proses *looping* global.

```
public struct Tpassword
{ public string desc,user,passw,note;
}
//jumlah total daftar password yang diperbolehkan
private const int PASSWORD_DATA = 500;
//path directory file password
private const string FILE_PASSWORD = "Program
Files/SmartDeviceApplication/psswrd.dat";
//deklarasi tipe data global
private Tpassword[] password = new
Tpassword[PASSWORD_DATA];//array password
private int idesc,i,j;//idesc =>index listbox yang dipilih, j =>total
daftar password, i => variabel looping global
private int[] id = new int[PASSWORD_DATA];
private bool list_id = false;
```

Inisialisasi sistem dengan mengisi *array password* dengan data yang ada dalam file “psswrd.dat”. Sebelumnya dicek terlebih dahulu apakah file”psswrd.dat” sudah ada atau belum jika belum maka proses inisialisasi pengisian tidak berjalan, sebaliknya jika file “psswrd.dat” sudah ada maka proses inisialisasi pengisian berlangsung

```
//membaca data dari file password dan disimpan di array password
private void read_file_password()
{
FileStream fs = new
FileStream(FILE_PASSWORD,FileMode.Open,FileAccess.Read);
BinaryReader br = new BinaryReader(fs);
for (int r=0;r<=(PASSWORD_DATA-1);r++)
{
if (br.PeekChar() != -1)//-1 => char tidak ketemu
{
j=r+1;
password[r].desc=br.ReadString();
password[r].user=br.ReadString();
password[r].passw=br.ReadString();
password[r].note=br.ReadString();
br.ReadString();//baca pembatas
} }
br.Close();fs.Close();
}
```

Variabel j diisi dengan jumlah data *password* yang ada dalam file “psswrd.dat” ditambah dengan nilai 1. Berikut fungsi-fungsi yang dibuat untuk memudahkan dalam penggunaan karena fungsi-fungsi tersebut dipanggil berulang kali.

```
//memasukan data ke array password
public void input_password(int n,Tpassword[] psswrd,string ds,string
us,string ps,string nt)
{
psswrd[n].desc=ds;psswrd[n].user=us;
psswrd[n].passw=ps;psswrd[n].note=nt;
}
```

Fungsi *input_password* digunakan untuk memasukkan data ke dalam *array password.*

```
//menulis data ke file password
private void write_file_password(int m,string fn,System.IO.FileMode
fm,System.IO.FileAccess fa)
{
FileStream fs = new FileStream(fn,fm,fa);
BinaryWriter bw = new BinaryWriter(fs);
bw.Write(password[m].passw);bw.Write(password[m].note);
//kode pembatas alt+0133
bw.Write("...");
bw.Close();fs.Close();
}
```

Fungsi *write_file_password* digunakan untuk menulis 1 data dalam *array* ke file "psswrd.dat". sedangkan untuk menulis semua data digunakan fungsi *write_all_file_password*.

```
//menulis semua data dari array password ke file password
private void write_all_file_password()
{
for (int w=0;w<=(PASSWORD_DATA-1);w++)
{
if (password[w].desc != null)
{
if (w == 0)
{write_file_password(w,FILE_PASSWORD,FileMode.CreateNew,FileAcc
ess.Write);}
Else
{write_file_password(w,FILE_PASSWORD,FileMode.Append,FileAccess
.Write);}
} } }
```

Untuk mengurutkan data yang masuk digunakan metode *insertion sort* dengan memanggil fungsi *search_pos_password*. Pengurutan data dilakukan pada waktu user menginputkan data. Jadi data akan teratur sesuai dengan urutan deskripsi. Fungsi

view_list_password digunakan untuk menampilkan data *array password* ke dalam tampilan *listbox* yang meperlihatkan deskripsi dan keterangannya.

3. Implementasi Pada Fitur *Task Schedule*

Deklarasi dan inisialisasi sistem dilakukan pertama kali saat program dijalankan. Deklarasi berupa jumlah daftar *task* yang diperbolehkan, path directory file "task.dat" untuk menyimpan daftar *task, array* utama untuk menyimpan data, variabel iname yang merupakan index listbox ketika dipilih, variable j yang merupakan total daftar untuk *array,* dan variable i untuk proses *looping* global.

```
public struct Tdatetime
{
public string Tdate,Ttime;
}
public struct Ttask
{
public string subject,note,uni,birthyear;
public byte priority,occurs;
public bool done,scheduled,s_alarm;
public Tdatetime due,alarm;
}
```

4. Implementasi Pada Aplikasi *Checking Task Schedule*

Deklarasi dan inisialisasi sistem dilakukan pertama kali saat program dijalankan. Deklarasi berupa jumlah daftar task yang diperbolehkan, path directory file "task.dat" untuk menyimpan daftar *task, array* utama untuk menyimpan data, variable j yang merupakan total daftar untuk array, dan variable i untuk proses *looping* global.

```
{
public string Tdate,Ttime;
}
public struct Ttask
{
public string subject,note,birthyear;
public byte priority,occurs;
public bool done,scheduled,s_alarm;
public Tdatetime due,alarm;
}
```

Fungsi AlarmOn digunakan untuk pengecekan sekaligus menampilkan informasi jika syarat-syarat seperti status alarm-nya true, status done-nya false. Lalu dilakukan pengecekan occurs. Jika occur-nya none, dicek apakah alarm date dan time nya sama dengan date dan time hari ini jika ya ditampilkan informasi kemudian dilakukan proses update terhadap data task yaitu status done dirubah menjadi true, jika occcur-nya tidak sama dengan none dicek apakah alarm date dan time nya sama dengan date dan time hari ini jika ya ditampilkan informasi. Fungsi ini dijalankan secara berkala dengan penggunaan timer.

5. Tampilan

Berikut ini adalah tampilan menu utama, yang menampilkan menu *contact, password, task,* dan *tool* pada sistem *organizer.*

Gambar 2. Tampilan Menu Utama

Tampilan pada *checking task schedule*, berfungsi ketika tanggal dan waktu sekarang telah mencapai tanggal dan waktu alarm maka akan muncul informasi bahwa ada kegiatan yang harus dilakukan dan daftar kegiatan tersebut tertera pada *listbox.*

Gambar 3. Informasi *checking task schedule*

KESIMPULAN

Berdasarkan dari hasil penelitian dan pengujian yang telah dilakukan, maka dapat ditarik kesimpulan bahwa :

- Perangkat lunak yang dikembangkan dapat berjalan pada PDA dengan sistem operasi Windows Pocket PC dan fitur yang ada sudah merupakan satu kesatuan.
- Fitur *task schedule* pada perangkat lunak yang secara otomatis dapat memanggil aplikasi pengecekan *task* melalui tombol yang disediakan.
- Dengan adanya sistem ini maka para user akan lebih mudah dalam melakukan aktivitasnya secara terencana.

DAFTAR PUSTAKA

1. Microsoft Corporation (2004). "Developer Resource For Windows Mobile 2003 Second Edition".

2. Gunarto, Hary (2004). "Introduction to Visual C++.Net and C#.Net" Penerbit Andi, Yogyakarta.

3. Kadir, Abdul (1995). "Pemograman C++" Penerbit Andi, Yogyakarta.

4. Kurniawan, Agus (2003). "Pemograman ADO.Net dengan C#" Penerbit Elex Media Komputindo, Jakarta.

5. Microsoft Corporation (2005). "MSDN Library April 2005".

6. Tan Soei Tien (2001). "Bahasa C# untuk Pemograman Berorientasi Objek" Penerbit Elex Media Komputindo, Jakarta.