



Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

BAB III

LANDASAN TEORI

3.1. *.Net Framework*

Microsoft .NET Framework (dibaca *Microsoft Dot Net Framework*) adalah sebuah komponen yang dapat ditambahkan ke sistem operasi *Microsoft Windows* atau telah terintegrasi ke dalam *Windows*. Integrasinya mulai dari *Windows Server 2003* dan versi-versi *Windows* terbaru. Kerangka kerja ini menyediakan sejumlah besar solusi-solusi program untuk memenuhi kebutuhan-kebutuhan umum suatu program baru, dan mengatur eksekusi program-program yang ditulis secara khusus untuk framework ini. *.NET Framework* adalah kunci penawaran utama dari *Microsoft*, dan dimaksudkan untuk digunakan oleh sebagian besar aplikasi-aplikasi baru yang dibuat untuk platform *Windows*.

Bahasa mesin (assembly) yang dihasilkan oleh kompiler **.NET** dinamakan sebagai “*Intermediate Language (IL)*”. Kenapa dinamakan sebagai bahasa mesin (Assembly) ? Bukankah assembly artinya bahasa yang dimengerti oleh mesin ? IL memang bahasa mesin namun bukan bahasa mesin untuk *CPU* nyata, melainkan bahasa mesin untuk *CPU Virtual* yang diciptakan oleh *Microsoft*. Jadi hanya *CPU Virtual* inilah yang memahami apa maunya *program .exe* yang telah dibuat oleh bahasa **.NET**.¹

¹<http://tts8189.wordpress.com/2009/03/30/apa-itu-net-framework/>

3.2. C#

C# merupakan sebuah bahasa pemrograman *component based* yang dikembangkan oleh Microsoft sebagai bagian dari inisiatif kerangka *.NET Framework*². Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek-aspek ataupun fitur bahasa yang terdapat pada bahasa-bahasa pemrograman lainnya seperti (*Java, Delphi, Visual Basic*, dan lain-lain) dengan beberapa penyederhanaan.

Pada akhir dekade 1990-an, Microsoft membuat program *Microsoft Visual J++* sebagai sebuah langkah percobaan untuk menggunakan *Java* di dalam sistem operasi Windows untuk meningkatkan antarmuka dari *Microsoft Component Object Model* (COM). Akan tetapi, akibat masalah dengan pemegang hak cipta bahasa pemrograman *Java, Sun Microsystems, Microsoft* pun menghentikan pengembangan J++, dan beralih untuk membuat pengganti J++, kompilernya dan mesin virtualnya sendiri dengan menggunakan sebuah bahasa pemrograman yang bersifat *general-purpose*. Untuk menangani proyek ini, Microsoft merekrut Anders Helsberg, yang merupakan mantan karyawan Borland yang membuat bahasa *Turbo Pascal*, dan *Borland Delphi*, yang juga mendesain *Windows Foundation Classes* (WFC) yang digunakan di dalam J++. Sebagai hasil dari usaha tersebut, C# pun pertama kali diperkenalkan pada bulan Juli 2000 sebagai sebuah bahasa pemrograman modern berorientasi objek yang menjadi sebuah bahasa pemrograman utama dalam pengembangan pada *platform Microsoft .NET Framework*.

²<http://ondotnet.com/pub/a/dotnet/2005/10/03/what-is-csharp.html>

Ada kritik yang menyatakan C# sebagai bahasa yang berbagi akar dari bahasa-bahasa pemrograman lain. Fitur-fitur yang diambilnya dari bahasa C++ dan Java adalah desain berorientasi objek, seperti *garbage collection*, *reflection*, akar kelas (*root class*), dan juga penyederhanaan terhadap pewarisan jamak (*multiple inheritance*). Fitur-fitur tersebut di dalam C# kini telah diaplikasikan terhadap iterasi, properti, kejadian (*event*), *metadata*, dan konversi antara tipe-tipe sederhana dan juga objek.

3.3 NHibernate

NHibernate adalah sebuah ORM (*Object-Relational Mapping*) *Open Source* untuk .Net Framework. NHibernate menyediakan kerangka untuk mapping model berbasis *object oriented* dengan *relational database*.³

NHibernate juga menyediakan fungsi untuk *query* dan fasilitas untuk pengambilan data. *Command SQL* sudah di-*generate* oleh NHibernate dan sangat memudahkan developer pada saat melakukan *coding*.

³<http://community.jboss.org/wiki/NHibernateforNET>

3.4 Desain *User Interface* Berbasis pada 8 *Golden Rules*

Desain pada *user interface* menjadi salah satu faktor penting dalam pengembangan suatu sistem. Dengan desain *interface* yang baik akan memberikan kemudahan dan kenyamanan pada seorang pengguna dalam mengoperasikan sistem. Salah satu aturan dalam mendesain *user interface* yang baik adalah 8 *Golden Rules* yang dicetuskan oleh Schneiderman.⁴

Berikut adalah 8 *Golden Rules* yang dikemukakan oleh Schneiderman, yaitu

a. **Konsistensi**

Konsistensi dilakukan pada urutan tindakan, perintah, dan istilah yang digunakan pada *prompt*, menu, serta layar bantuan.

b. **Feedback yang jelas**

Diperlukan feedback yang baik agar pengguna mengetahui hasil dari interaksi yang dilakukannya. Dengan adanya feedback akan memudahkan pengguna untuk mengerti kerja dari program. Misalnya muncul suatu suara ketika salah menekan tombol pada waktu input data atau muncul pesan kesalahannya.

c. **Mencegah Kesalahan**

Desain dari sistem haruslah dapat mencegah pengguna melakukan kesalahan. Jika sampai user melakukan kesalahan, sistem tersebut dapat mengembalikan ke keadaan sebelum *error* tersebut terjadi.

⁴<http://iwanbinanto.wordpress.com/2008/07/21/8-aturan-emas-desain-user-interface/>

d. **Mudah kembali ke tindakan sebelumnya**

Adanya kemudahan untuk kembali ke tindakan sebelumnya membuat pengguna lebih bebas untuk mengeksplorasi sistem.

e. **Mengurangi beban ingatan jangka pendek**

Pemberian simbol atau gambar yang jelas disertai urutan yang tidak berubah-ubah akan memudahkan pengguna untuk mengingat langkah-langkah sistem dioperasikan.

f. **Memungkinkan pengguna untuk menggunakan *shortcut***

Ada kebutuhan dari pengguna yang sudah ahli untuk meningkatkan kecepatan interaksi, sehingga diperlukan singkatan, tombol fungsi, perintah tersembunyi, dan fasilitas makro.

g. **Merancang dialog untuk menghasilkan suatu penutupan**

pengaturan urutan aksi dari awal sampai akhir dan diakhiri dengan feedback.

h. **Mendukung tempat pengendali internal (*internal locus of control*)**

Pengguna ingin menjadi pengontrol sistem dan sistem akan merespon tindakan yang dilakukan pengguna daripada pengguna merasa bahwa sistem mengontrol pengguna.