BAB III

METODOLOGI

3.1 Metode Analisis

Dalam proses pengembangan sebuah sistem perangkat lunak, sebelum kita memasuki tahap perancangan sistem dan tahap pengkodean (implementasi) kita diharuskan melakukan analisa terhadap sistem yang akan dibuat. Analisis diperlukan agar sistem yang dibuat kelak dapat menjadi sebuah sistem yang memenuhi dan sesuai kebutuhan. Analisis sistem adalah sebuah tehnik pemecahan masalah dengan menguraikan sebuah sistem secara utuh menjadi bagian-bagian komponen penyusun sistem yang lebih kecil, guna mempelajari kinerja dari komponen-komponen dalam sistem tersebut.

Metode analisis yang digunakan dalam pengembangan perangkat lunak pada penelitian ini adalah metode analisis berorientasi objek (*Object Oriented Analysis*). Analisis berorientasi objek (*Object Oriented Analysis*) adalah tahapan perangkat lunak dengan menentukan SRS (*Sytem Requirement Spesification*), identifikasi kelas-kelas serta hubungannya satu terhadap yang lain [NUG05]. Dalam hal ini kita harus melihat sistem sebagai kumpulan objek-objek yang saling berinteraksi.

Aplikasi *game puzzle rubik cube* merupakan versi komputer implementasi dari permainan *puzzle rubik cube* yang sudah dikenal selama ini, dimana para pemain ditantang untuk menyamakan seluruh sisi kubus yang terdiri dari enam sisi warna yang berbeda dengan cara memutar setiap sisinya, yang sebelumnya ke

enam sisi tersebut diputar secara acak. Untuk menyamakan seluruh sisi tersebut pemain dapat memutar seluruh sisi kubus baik searah maupun berlawanan arah dengan arah jarum jam. Tidak ada batasan waktu dalam menyelesaikan *game* ini, pemain dapat bermain sampai selesai maupun memutuskan untuk menyerah.

3.2 Hasil Analisis

Berdasarkan deskripsi dan teori-teori pada bagian terdahulu maka didapatkan beberapa hasil analisis kebutuhan yang berorientasi objek, antara lain:

3.2.2 Identifikasi Aktor

Aktor adalah orang atau sistem yang berhubungan dengan sistem serta berada diluar sistem. Dari definisi tersebut didapatkan aktor sebagai berikut:

- **Aktor** pada *game* ini adalah **pemain** (yang bermain *game puzzle rubik cube).*

3.2.3 Identifikasi *Use case*

Use case dapat didefinisikan sebagai hal-hal atau tindakan-tindakan yang dilakukan oleh aktor pada sistem. Dapat dianalisis beberapa use case pada aplikasi ini, antara lain:

- *Use case* **pemanggilan game**, pada *use case* ini pemain memanggil aplikasi untuk dijalankan.
- *Use case* ***new game***, pada *use case* ini pemain memulai permainan baru.
- *Use case* ***options***, pada *use case* ini pemain menentukan pengaturan pada permainan. Mencakup menentukan tekstur *rubik*, dan kecepatan animasi putaran sisi *rubik.*

- *Use case* ***besttime***, pada *use case* ini pemain dapat melihat catatan rekor waktu yang ada.
- *Use case* ***help***, pada *use case* ini pemain membuka file bantuan yang ada.
- *Use case* ***about***, pada *use case* ini pemain membuka form about yang berisi keterangan tentang pembuat *game*.
- *Use case* ***exit***, pemain menutup dan mengakhiri aplikasi *game*.

3.2.4 Identifikasi Objek-objek

Dalam menentukan objek-objek pada sebuah sistem hendaknya menggunakan kata benda sebagai basis untuk penentuan kelas atau objek [NUG05]. Dapat dianalisis objek-objek antara lain:

- Cubik (tersusun atas dua puluh tujuh kubus).
- Kubus (tersusun atas enam sisi atau enam buah persegi).
- Sisi (tersusun atas empat titik).
- Titik (tersusun atas koordinat : x,y dan z).
- *Window* atau Jendela tempat menampilkan hasil render Rubik pada layar monitor.

3.2.5 Identifikasi Metode atau Fungsi

Pada *game* ini dibutuhkan beberapa metode agar *game* dapat berjalan dengan baik, beberapa metode yang dapat dianalisis antara lain:

- Menggambar rubik, metode ini akan menjalankan metode penggambaran kubus sebanyak dua puluh tujuh kali.
- Menggambar kubus, metode ini akan menjalankan metode penggambaran sisi sebanyak enam kali.

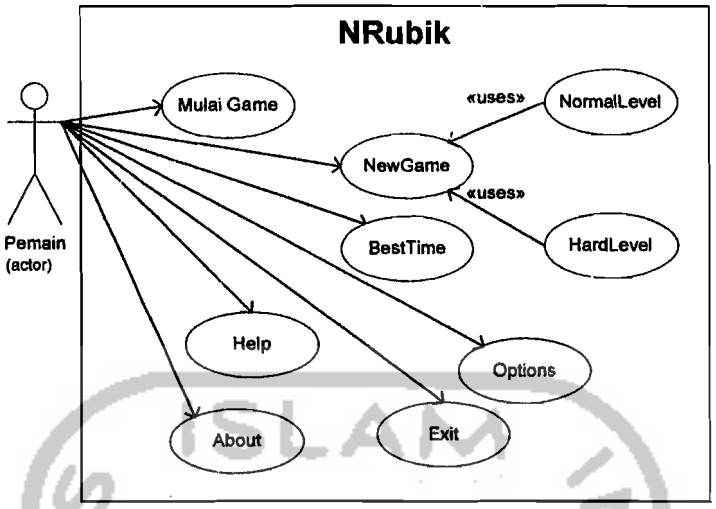
- Menggambar sisi, metode ini akan menggambar sebuah persegi (empat buah titik).
- Menentukan koordinat, metode ini akan memberikan koordinat awal posisi titik-titik (*verteks*) pada keempat titik sisi-sisi rubik.
- Memutar sisi, metode ini akan memutar sisi-sisi rubik.
- Mengalikan koordinat awal dengan rumus rotasi agar koordinat titik-titik pada sisi rubik dapat berubah, sehingga sisi rubik dapat berputar.

3.3 Metode Perancangan

Setelah mendapatkan kebutuhan perangkat lunak pada tahap analisis, maka kita dapat melakukan perancangan sebelum masuk ke tahap implementasi. Beberapa tahap perancangan yang akan dilaksanakan dengan membuat diagram-diagram UML antara lain:

3.3.1 Diagram Use case

Diagram *use case* menunjukkan interaksi antara sistem dengan aktor ataupun sistem eksternal lainnya. Berdasarkan pada analisis yang telah dilakukan maka dapat digambarkan diagram *use case* sebagai berikut:



Gambar 3.1 Diagram *Use Case*

3.3.2 Diagram Kelas

Diagram kelas menggambarkan perangkat lunak secara statis, beserta hubungan (relasi-relasi) kelas yang satu dengan kelas-kelas lainnya. Disini juga digambarkan operasi-operasi yang di lakukan sebuah kelas beserta atribut-atributnya. Kelas-kelas yang digunakan pada pembuatan *game* Nrubik diantaranya:

a. Kelas CNRubikDlg

Kelas ini adalah kelas yang digunakan sebagai tampilan utama (form menu utama) pada kelas ini terdapat metode-metode yang berkaitan dengan menu-menu yang akan kita pilih (misal: metode OnMenuAbout yang akan dijalankan jika kita memilih menu *about* pada menu utama atau kelas ini, OnMenuHelp yang akan dijalankan jika kita memilih menu *help,* OnMenuTimer yang akan menampilkan catatan waktu permainan pada menu utama ini dan lain-lain).

b. Kelas COpenGL

Kelas ini adalah kelas yang digunakan sebagai "kanvas" bagi obyek OpenGL yang akan kita buat atau kita gambar, objek dari kelas ini kelak akan ditempelkan pada objek dari kelas CNRubikDlg (form utama). Pada kelas ini juga menangani objek cubik (rubik) sebagai sebuah objek utuh (terdiri dari dua puluh tujuh kubus).

c. Kelas RKubus

Kelas ini adalah kelas yang akan menangani objek kubus (yang terdiri atas enam buah sisi).

d. Kelas RSisi

Kelas ini adalah kelas yang akan menangani objek sisi (yang terdiri atas empat buah titik pada setiap sisinya).

e. Kelas RTitik

Kelas ini adalah kelas yang merupakan realisasi dari keseluruhan objek Rubik (cubik), karena kelas inilah yang sesungguhnya yang menyusun seluruh objek dengan koordinat-koordinat 3D (x, y, z).

f. Kelas CBestTimeDlg

Kelas ini adalah kelas yang akan menampilkan form waktu rekor (*record time*).

g. Kelas COptionsDlg

Kelas ini adalah kelas yang akan menampilkan form pilihan seting *game* (*options*). Kelas ini akan memberikan pilihan pengubahan tekstur maupub kecepatan putar rubik.

h. Kelas CAboutDlg

Kelas ini adalah kelas yang akan menampilkan form tentang (*about*).

i. Kelas CFillRecDlg

Kelas ini adalah kelas yang akan menampilkan form untuk pengisian nama bagi pemecah waktu rekor (*record time*).

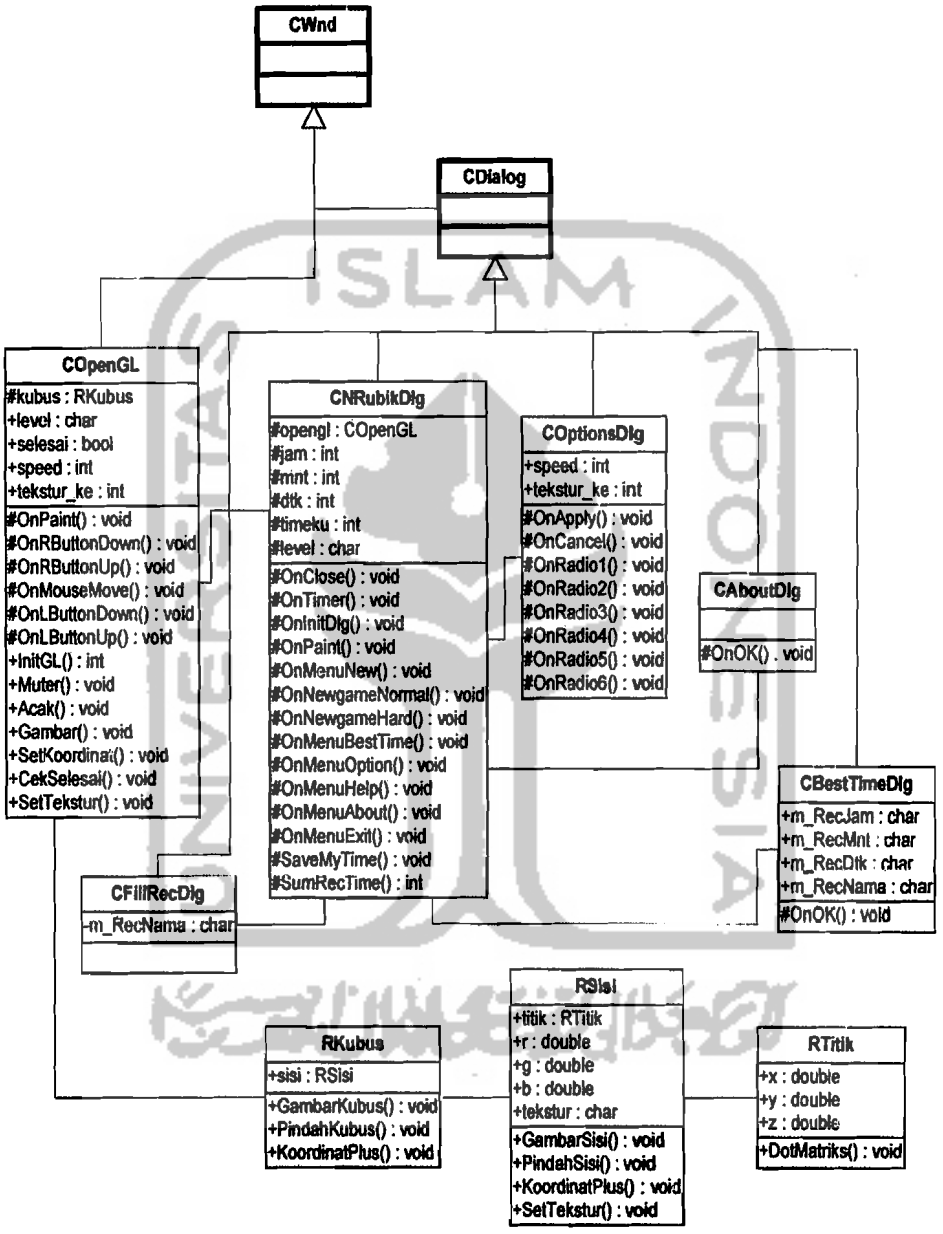
j. CWnd

Kelas bawaan Visual C++ (MFC) yang menangani aplikasi window.

k. CDialog

Kelas bawaan Visual C++ (MFC) yang digunakan sebagai pembentuk form (dialog). Kelas ini akan menjadi turunan bagi kelas-kelas dialog (form-form) yang akan diciptakan pada sebuah aplikasi MFC yang berbasis dialog.

Berdasarkan pada analisis yang telah dilakukan maka dapat digambarkan diagram kelas seperti pada gambar 3.2 berikut :



Gambar 3.2 Diagram Kelas

\# : **protected** (hanya dapat diakses kelas tersebut beserta turunannya).

\+ : **public** (dapat diakses seluruh kelas).

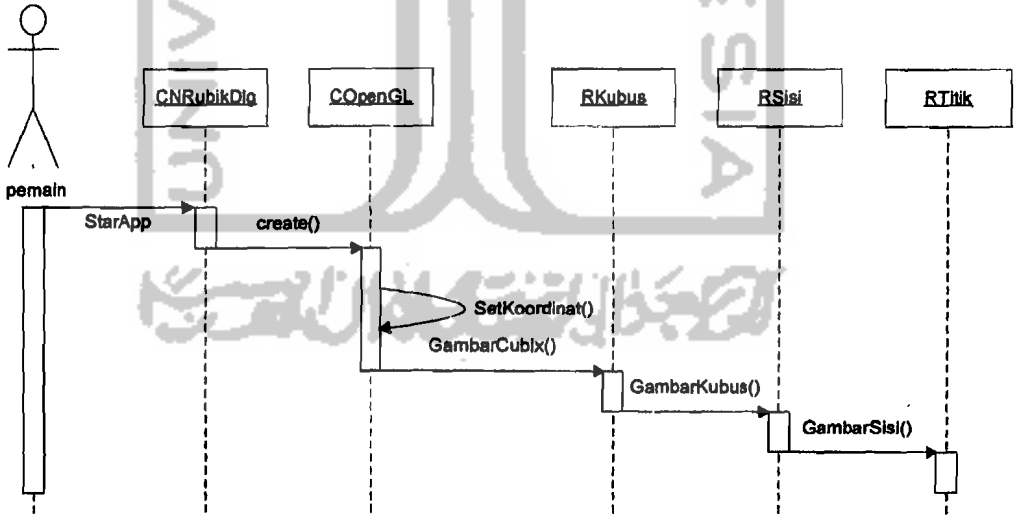
Kelas **CWnd** dan **CDialog** (cetak tebal), adalah kelas bawaan Visual C++.

3.3.3 Diagram Sekuen

Diagram sekuen atau yang sering disebut *sequence diagram* adalah salah satu diagram UML yang memodelkan logika sebuah *use case*, dengan kata lain diagram ini menggambarkan sebuah *use case*. Diagram sekuen menggambarkan bagaimana objek berinteraksi dengan satu sama lain melalui pesan pada eksekusi sebuah *use case* [WHI04]. Diagram ini terdiri atas dimensi vertikal (atas-bawah) yang menggambarkan waktu atau masa aktif objek, sedangkan dimensi horisontal (datar) menggambarkan pesan (interaksi) antara objek-objek terkait.

1. Diagram Sekuen Mulai*Game*

Diagram sekuen untuk *use case* Mulai*Game* dapat dilihat pada Gambar 3.3 sebagai berikut :

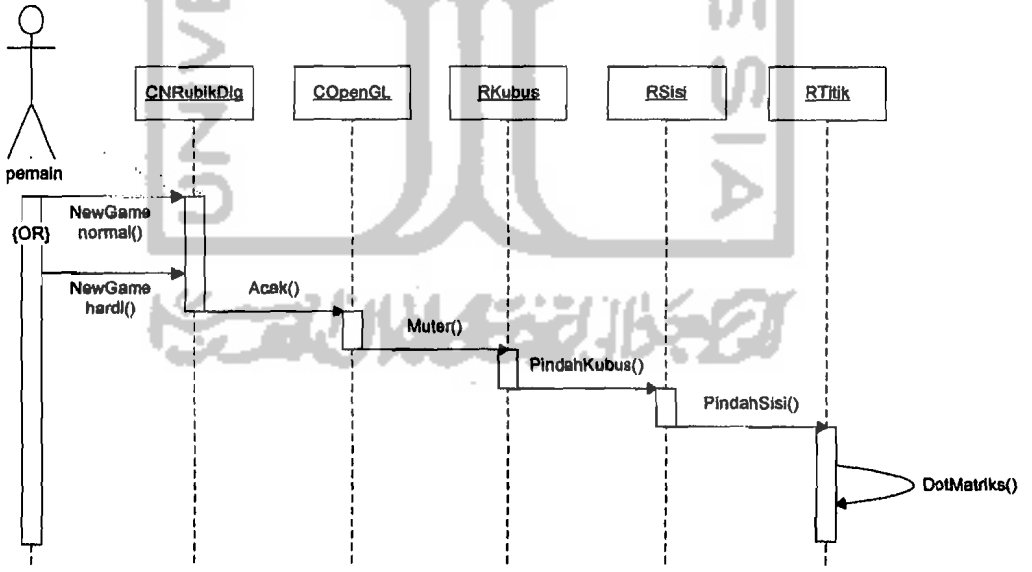


Gambar 3.3 Diagram Sekuen Mulai*Game*

Dari gambar diatas dapat dilihat bahwa ketika pemain menjalankan aplikasi untuk pertama kali maka kelas yang dipanggil pertama adalah kelas CNRubikDlg kelas tersebut akan memunculkan *main form* (form utama) dari game ini, kemudian kelas tersebut akan menciptakan objek COpenGL. Kelas COpenGL sendiri akan mengeset koordinat lalu memanggil fungsi GambarCubik dan memanggil kelas RKubus yang akan memanggil fungsi gambar kubus dan seterusnya hingga tergambar sebuah rubik cube utuh.

2. Diagram Sekuen *NewGame*

Diagram sekuen untuk *use case NewGame* dapat dilihat pada Gambar 3.4 sebagai berikut :



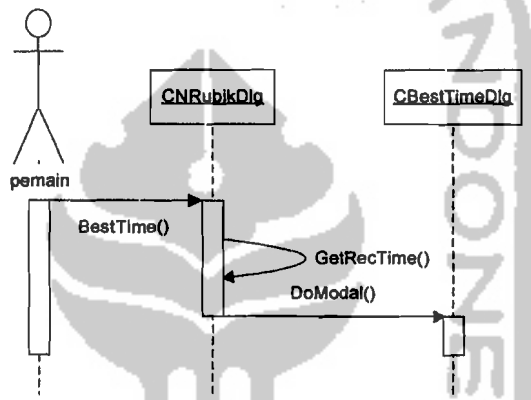
Gambar 3.4 Diagram Sekuen *NewGame*

Pada gambar diatas dapat dilihat pemain dapat memilih permainan baru dengan level normal atau hard. Lalu sistem berturut-turut akan melakukan

pengacakan terhadap koordinat rubik dengan menjalankan metode-metode yang mendukungnya.

3. Diagram Sekuen *BestTime*

Diagram sekuen untuk *use case BestTime* dapat dilihat pada Gambar 3.5 sebagai berikut :

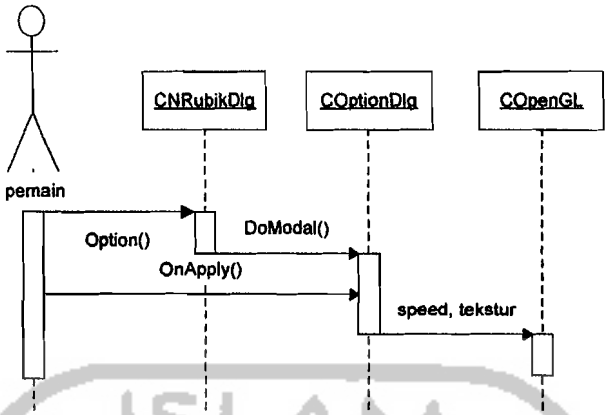


Gambar 3.5 Diagram Sekuen *BestTime*

Pada gambar diatas pemain membuka menu *BestTime (record time),* kelas NRubikDlg (menu utama) akan meload catatan waktu rekor yang ada lalu menampilkan form *best time (record time).*

4. Diagram Sekuen *Options*

Diagram sekuen untuk *use case Options* dapat dilihat pada Gambar 3.6 sebagai berikut :

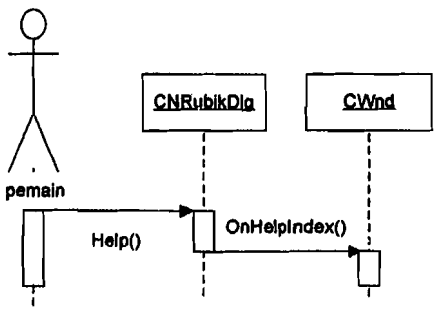


Gambar 3.6 Diagram Sekuen *Options*

Pada gambar diatas pemain membuka menu *Options* yang kemudian kelas NRubikDlg (menu utama) akan menampilkan form *option* (kelas COptionDlg) setelah menu *options* tampil pemain bisa memilih setting tekstur dan kecepatan. Pada akhirnya settingan pengguna akan di transfer ke kelas COpenGL untuk disetting ke *game*.

5. Diagram Sekuen *Help*

Diagram sekuen untuk *use case Help* dapat dilihat pada Gambar 3.7 sebagai berikut :

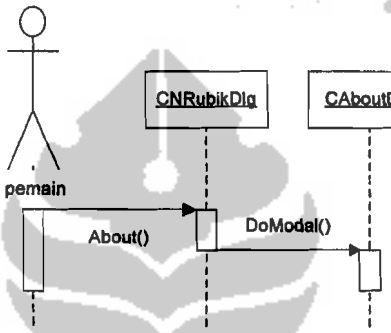


Gambar 3.7 Diagram Sekuen *Help*

Pada gambar diatas pemain membuka menu *Help* (bantuan), kelas NRubikDlg (menu utama) akan meminta metode *OnHelpIndex* dari kelas CWnd.

6. Diagram Sekuen *About*

Diagram sekuen untuk *use case About* dapat dilihat pada Gambar 3.8 sebagai berikut :

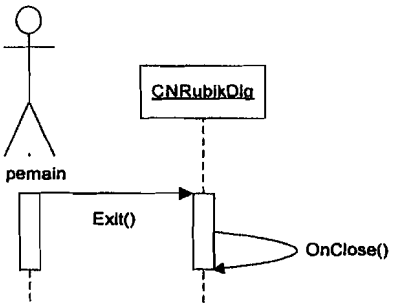


Gambar 3.8 Diagram Sekuen *About*

Pada gambar diatas pemain membuka menu *About* (tentang), kelas NRubikDlg (menu utama) akan menampilkan form *about* (kelas CaboutDlg).

7. Diagram Sekuen *Exit*

Diagram sekuen untuk *use case Exit* dapat dilihat pada Gambar 3.8 sebagai berikut :



Gambar 3.9 Diagram Sekuen *Exit*

3.3.4 Perancangan Antarmuka

1. Jendela Utama (CNRubikDlg)

Perancangan pada jendela utama memiliki dua menu utama dapat dilihat hierarkinya sebagai berikut:

▶ Game ▶▶ New game ▶▶▶ normal (memulai game level normal).

▶▶▶ hard (memulai game level hard).

▶ RecordTime (melihat catatan waktu terbaik).

▶ Options (memilih pengaturan *game*).

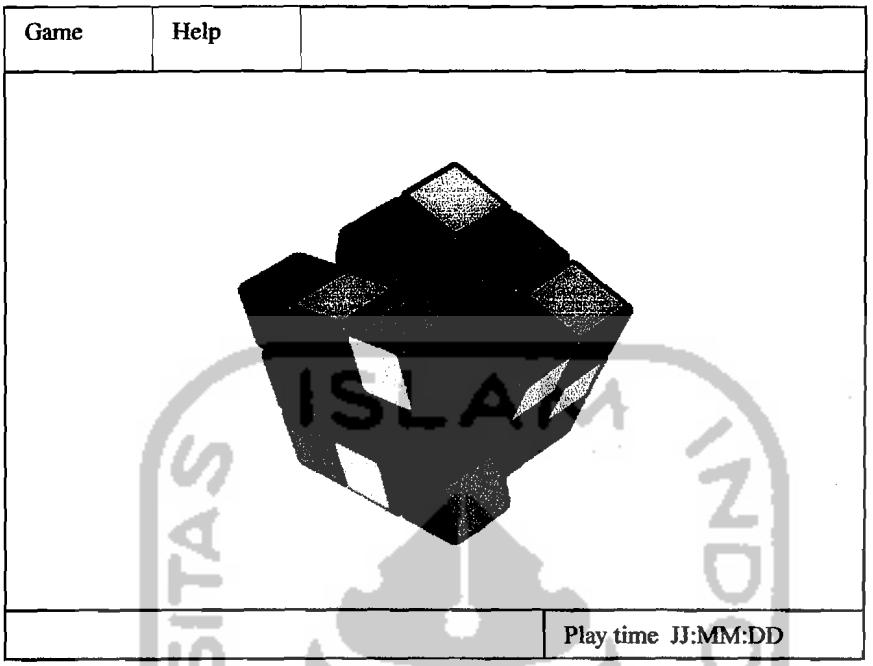
▶ Exit (keluar permainan).

▶ Help ▶ Help (membuka file bantuan).

▶ About (membuka file tentang).

Keterangan : ▶ Menu ▶ SubMenu ▶▶ Sub SubMenu

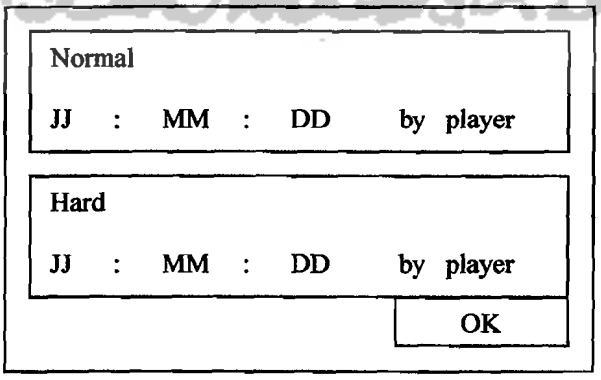
Gambar perancangan dari form menu utama (kelas CNRubikDlg) dapat dilihat pada gambar 3.10 berikut:



Gambar 3.10 Rancangan Form Menu Utama

2. *Record time (BestTime)*

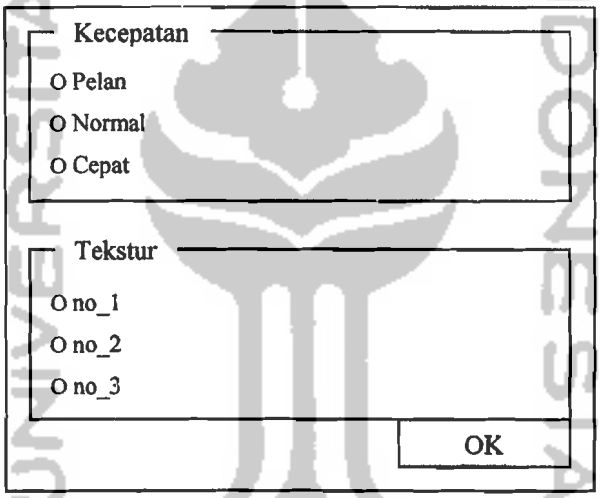
Pada antarmuka form *record time* memiliki dua tampilan catatan waktu rekor, yaitu catatan rekor level normal dan catatan waktu rekor level hard, masing masing catatan rekor memiliki format (jam : mnt : dtk). Perancangan antarmuka form *record time* dapat dilihat pada gambar 3.11 berikut :



Gambar 3.11 Rancangan Form *Record Time*

3. *Option*

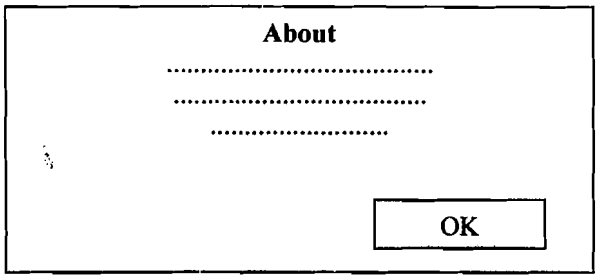
Pada antarmuka form *options* memiliki dua pilihan setingan pada *game,* yaitu seting kecepatan putar sisi rubik dan seting pilihan tekstur. Pada seting kecepatan menyediakan tiga pilihan kecepatan putar (pelan, normal, cepat) yang masing-masing dapat dipilih dengan *radio box*. Pada setingan tekstur memiliki tiga pilihan tekstur yang dipilih juga menggunakan radio box. Perancangan antarmuka form *options* dapat dilihat pada gambar 3.12 berikut :



Gambar 3.12 Rancangan Form Options

4. *About*

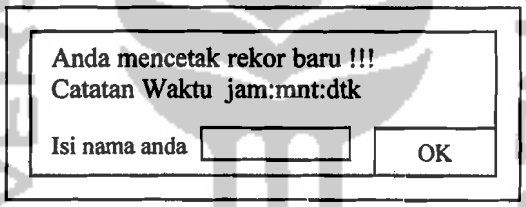
Form ini digunakan untuk melihat catatan about (tentang) *game* ini baik pembuat tahun pembuatanya dan data-data lainnya. Perancangan antarmuka formnya dapat dilihat pada gambar 3.13 berikut :



Gambar 3.13 Rancangan Form About

5. *Isi Pencetak Rekor*

Form ini digunakan jika kita menyelesaikan permainan dengan catatan waktu yang lebih baik dari catatan waktu yang sudah ada. Perancangan antarmuka formnya dapat dilihat pada gambar 3.14 berikut :



Gambar 3.14 Rancangan Form Isi Pencetak Rekor