BAB III

ANALISIS DAN PERANCANGAN SISTEM

III.1 Analisis Sistem

Pada bab ini akan dibahas mengenai analisis dan perancangan pada sistem yang dibangun.

III.1.1 Analisis Masalah

Analisis masalah adalah suatu gambaran masalah yang diangkat dalam penulisan skripsi tentang penerapan algoritma minimax untuk permainan nim. Algoritma ini digunakan untuk mencari langkah yang tepat bagi komputer untuk memperoleh kemenangan.

Berikut adalah hal-hal yang akan dianalisis berkaitan dengan penerapan langkah kerja minimax dalam permainan nim :

1. Analisis komponen permainan
2. Analisis strategi bermain
3. Langkah kerja minimax

III.1.2 Analisis Permainan

Dalam beberapa *game playing* yang memiliki karakteristik permainan dengan ciri dimainkan oleh 2 pemain, dilakukan secara bergiliran, kerugian seorang pemain merupakan keuntungan pemain lainnya, dan tidak mengandung

probabilistik seperti permainan yang menggunakan dadu. Maka minimax merupakan metode yang cocok untuk diterapkan.

Berbeda dengan permainan lain, nim merupakan permainan logika matematika yang sangat sederhana namun jika kita perdalam lagi maka akan kita temukan strategi bermain nim yang cukup menarik. Dalam penulisan tugas akhir ini ada beberapa data dijadikan sebagai bahan penelitian yaitu data mengenai komponen permainan, strategi dalam bermain, dan data mengenai langkah kerja algoritma minimax.

III.1.2.1 Analisis Komponen Permainan

1. Identifikasi Pemain

Permainan nim hanya dapat dimainkan oleh dua pemain. Masing-masing pemain bermain secara bergiliran.

2. Cara Bermain

Sama seperti nama permainannya yaitu *nimm* dalam bahasa jerman yang artinya “ambil” atau istilah dalam bahasa cinanya yaitu *“Tsyanshidzi”* (mengambil batu). Umumnya cara bermainnya adalah dengan mengambil sebuah atau beberapa benda dalam tumpukan (baris) yang sudah ditentukan. Permainan terus dilakukan secara bergiliran hingga jumlah benda dalam tumpukan habis.

3. Aturan Permainan

Pada faktanya bentuk permainan nim sangat variatif. Ada yang dimainkan hanya dalam satu tumpukan (baris) dan adapula yang dimainkan dalam beberapa tumpukan baris dengan masing-masing baris memiliki jumlah objek yang tidak

ditentukan berapa batas maksimumnya. Untuk menentukan siapa pemenangnya sendiri ada yang ditentukan berdasarkan standar konvensi permainan normal yang artinya pemain yang mengambil sisa benda terakhir adalah pemenangnya atau sebaliknya, pemain yang berhasil menyisakan benda untuk pemain lain adalah pemenangnya (misere).

4. Permainan Usai

Permainan dikatakan usai apabila tidak ada benda (objek) yang tersisa dalam tumpukan atau habis.

III.1.2.2 Analisis Strategi Permainan

Berikut akan dijelaskan bagaimana gambaran dari permainan nim. Agar lebih mudah dipahami, kita coba asumsikan dengan sebuah contoh kasus untuk permainan nim apabila ada satu tumpukan (baris) dengan jumlah bendanya (objek) sebanyak 10 buah.



Gambar III.1 Representasi 10 buah objek

Pada penjelasan sebelumnya sudah diterangkan tentang bagaimana memainkan permainan ini. Setiap pemain boleh mengambil satu, dua atau beberapa objek. Bagaimana caranya agar dapat memenangkan permainan ini.Kira-kira apa yang harus dilakukan? Berapa banyak objek yang harus diambil? jika

peraturannya adalah pemain yang mengambil benda (objek) terakhir adalah pemenangnya.

Artinya ketika memainkan permainan ini, dibutuhkan strategi untuk dapat menjadi pemenangnya. Apabila diasumsikan kita sebagai pemain pertama. Berapa banyak objek yang akan kita ambil? Berikut akan coba di gambarkan bagaimana cara bermainnya.

Misal pemain pertama, mengambil 3 buah objek terlebih dahulu. Maka akan tersisa 7 buah objek. Lihat gambar III.2.



Gambar III.2 representasi sisa 7 buah objek

Saat giliran pemain kedua dari beberapa sisa objek yang ada, diasumsikan dia mengambil 5 objek, maka objek yang tersisa tinggal 2 buah. Lihat gambar III.3

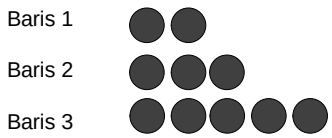


Gambar III.3 representasi sisa 2 buah objek

Giliran kita bermain kembali dengan sisa 2 buah objek. Agar memperoleh kemenangan maka kita dapat ambil atau memindahkan semua sisa objek tersebut sehingga tidak tersisa satu objek sama sekali. Dengan demikian dalam peraturan permainan normal, kita adalah pemenangnya. Karena biasanya permainan nim ini dimainkan dalam berbagai versi, permainan versi normal maksudnya pemain yang

terakhir mengambil objek dia sebagai pemenangnya. Sedangkan versi lainnya yaitu *misere*, merupakan kebalikan dari permainan normal yang membedakan adalah status pemenangnya.

Pada prinsipnya, dasar dari permainan nim adalah dengan menggunakan *binary digital sum* atau yang lebih dikenal dengan *exclusive or (xor)* atau *nim-sum.* Untuk beberapa tumpukan baris maka kita dapat melakukan perhitungan *nim-sum* nya. Sebagai contoh kasus lainnya apabila ada 3 tumpukan baris dengan masing-masing tumpukan berjumlah 2, 3, 5. Direpresentasikan pada gambar III.4.



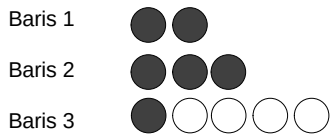
Gambar III.4 Nim dengan Tumpukan 3 Baris

Bagaimana strategi memenangkan permainannya? Dengan asumsi bahwa kita sebagai pemain ke-1. Kemudian cari *nim-sum* dari semua tumpukan dengan cara mengkonversi jumlah objek pada tiap tumpukan baris ke bilangan biner dan menjumlahkannya dengan menggunakan *exclusive or (xor).* Diperlihatkan pada tabel III.1

Tabel III.1 Representasi Perhitungan Nim Sum

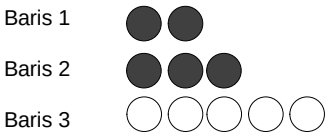
Tumpukan (baris)	**Jumlah objek (desimal)**	**Jumlah objek (biner)**	**Jumlah objek(Biner) xor *nim-sum***	**Hasil penjumlahan**
1	2	010	010 ⊕ 100 = 110	6
2	3	011	011 ⊕ 100 = 111	7
3	5	101	101 ⊕ 100 = 001	1
Nim-sum	2 ⊕ 3 ⊕ 5 = 4	100		

Dari penjumlahan biner pada tabel III.1, kita lihat bahwa satu-satunya baris yang berkurang jumlah objeknya adalah baris ke-3. Maka kita harus menyisakan 1 buah objek pada baris ke-3 yaitu dengan mengambil 4 objek dari baris ke-3. Lihat gambar III.5



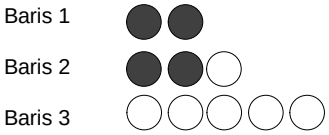
Gambar III.5 Representasi Nim dengan Sisa Objek 2, 3, 1 perbaris

Giliran pemain ke-2, sisa objek pada baris 1, 2, dan 3 adalah 2, 3, dan 1. Kita asumsikan pemain ke-2 mengambil 1 pada baris ke-3 sehingga menyisakan 2 dan 3 buah objek pada baris ke-1 dan 2.



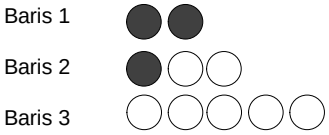
Gambar III.6 Representasi Nim dengan Sisa objek 2, 3 perbaris

Giliran pemain 1, apabila hanya tersisa 2 baris, maka langkah yang harus diambil adalah dengan mengambil objek pada baris yang lebih besar untuk membuat kedua tumpukan berjumlah sama. Yaitu dengan mengambil 1 buah objek pada baris ke-2.



Gambar III.7 Representasi Nim dengan Sisa Objek 2, 2 perbaris

Giliran pemain ke-2, misal pemain 2 mengambil 1 pada baris ke-2, maka sisanya diperlihatkan pada gambar III.8.



Gambar III.8 Representasi Nim dengan Sisa Objek 2, 1 perbaris

Giliran pemain ke-1 kembali. Hanya mengambil 1 objek pada baris ke-1. Selanjutnya apapun langkah yang lawan ambil, maka kita yang akan menang.

Adapun perbedaan strategi permainan nim biasa dengan permainan *misere* yaitu ketika hanya tesisa 2 tumpukan baris maka langkah yang harus diambil adalah dengan menyisakan tumpukan baris yang jumlah objeknya ganjil. Berarti pemain ke-1 harus mengambil semua objek pada baris ke-1, sehingga menyisakan 1 objek pada baris ke-2. Maka pemain ke-2 lah yang akan mengambil sisa objek tersebut dan ia kalah.

III.1.3 Analisis Metode Minimax

Dalam minimax ada dua prosedur yang harus dijalankan yaitu maximasi dan minimasi. Untuk memperjelas bagaimana penerapan dari minimax ini. Akan langsung digambarkan dengan *tree.*

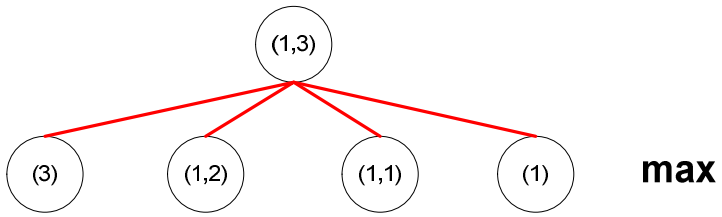
Pertama kita asumsikan pemain 1 sebagai max. Apabila diberikan 2 tumpukan baris dengan masing-masing tumpukan berjumlah 1 dan 3 buah objek. Maka kita dapat inisialisasikan baris tersebut dengan jumlah objeknya dengan *state* (1,3). *State* (1,3) ini kita asumsikan sebagai *root* node. Sebagai pemain pertama, max dapat melakukan beberapa kemungkinan jalan. Perlu diperhatikan, pada permainan nim tidak diperbolehkan untuk mengambil beberapa buah objek pada 2 atau lebih tumpukan baris sekaligus. Dengan demikian kemungkinan jalan yang dapat dilalui oleh max adalah sebagai berikut:

Mengambil 1 buah objek dari tumpukan pertama, sehingga menyisakan 3 buah objek pada tumpukan ke-2. Kita inisialkan dengan *state* (3), atau

1. Mengambil 1 buah objek pada tumpukan ke-2, sehingga menyisakan 1 pada tumpukan pertama dan 2 buah objek pada baris ke-2, (1,2). Atau,

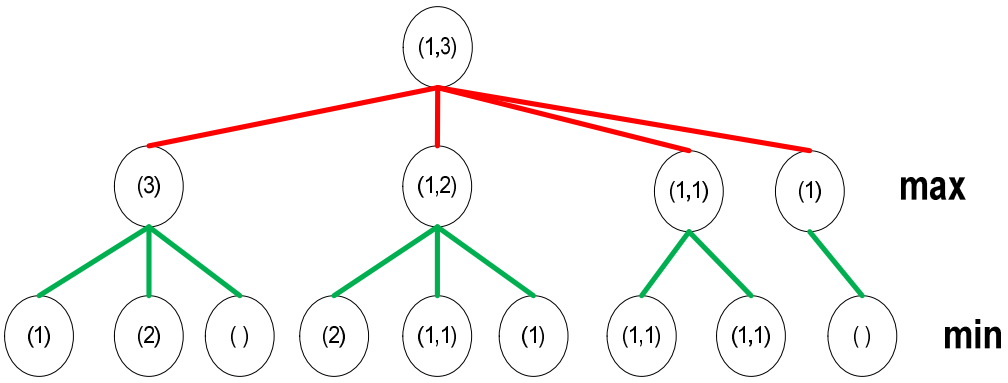
2. Mengambil 2 buah objek pada tumpukan ke-2, sehingga menyisakan 1 buah objek pada tumpukan pertama dan 1 buah objek pada tumpukan ke-2, (1,1). Atau,
3. Mengambil seluruh objek pada tumpukan ke-2, sehingga akan menyisakan 1 pada tumpukan pertama. Kita inisialkan (1).

Apabila digambarkan dalam *tree* maka hasilnya dapat diperlihatkan pada gambar III.9.



Gambar III.9 T*ree* Representasi Pergerakan Max

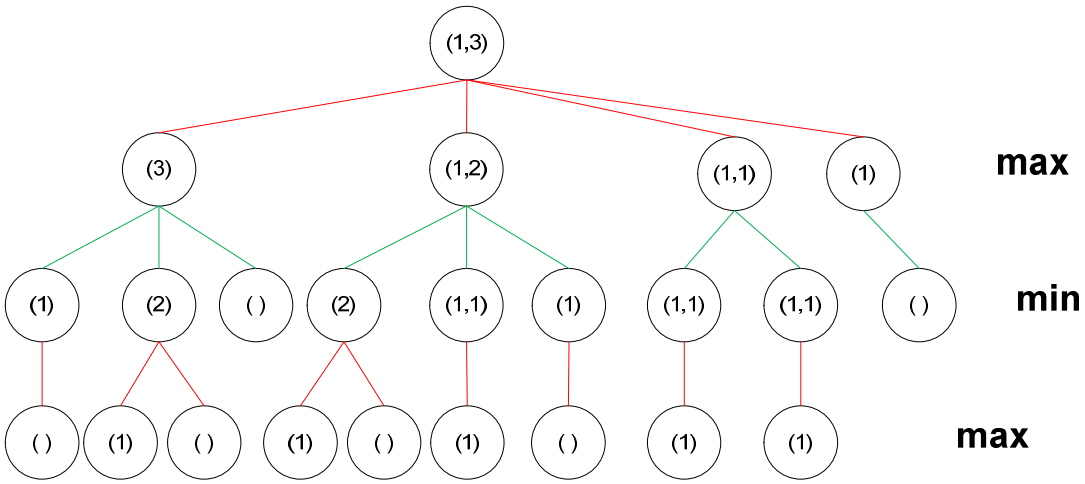
Kemudian pemain ke-2, kita asumsikan sebagai min. Dari langkah yang dibuat oleh max, maka min dapat membuat langkahnya yang merupakan hasil turunan dari setiap *node* yang ada.



Gambar III.10 T*ree* Langkah Pergerakan Max dan Min

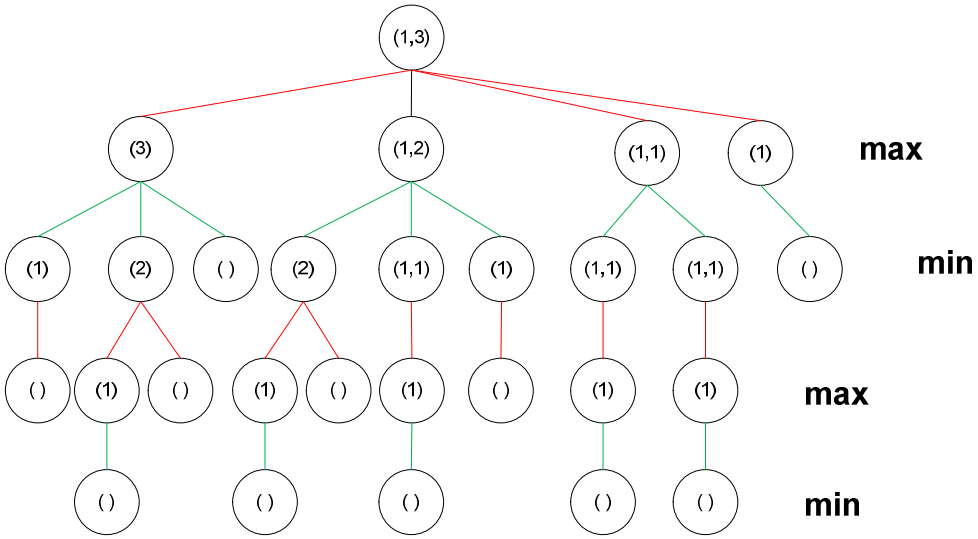
Pada gambar III.10. Digambarkan bahwa min dapat melakukan 3 langkah dari *state* (3), yaitu dengan mengambil 2 buah objek, sehingga menyisakan 1 buah objek, *state* (1). Atau min dapat mengambil 1 buah objek, sehingga menyisakan 2, *state* (2). Atau min mengambil semua objek yaitu 3 buah objek dari tumpukan, sehingga untuk tipe permainan normal min akan memperoleh kemenangan. Dan seterusnya untuk masing-masing *state* (1,2), (1,1) dan (1).

Dilanjutkan dengan giliran max. Max dapat konversi *state* (1) menjadi ( ). *State* (2) menjadi (1) dan ( ). *State* (1,1) menjadi (1). Dan (1) menjadi ( ). Jika max mengambil langkah yang menghasilkan *state* ( ), maka max menang. Lihat gambar III.11



Gambar III.11 T*ree* Langkah Max Selanjutnya

Giliran min, dari semua *state* yang masih menghasilkan kemungkinan jalan hanya tersisa satu buah objek, artinya min akan memperoleh kemenangan. Lihat gambar III.12



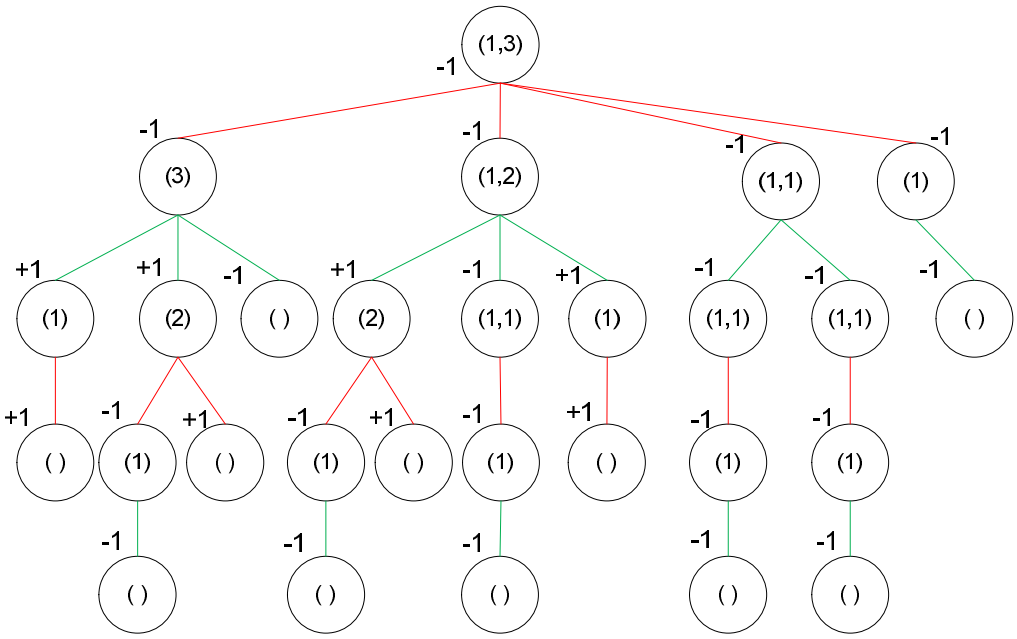
Gambar III.12 T*ree* Seluruh Kemungkinan Langkah Yang Dilalui

Dari struktur *tree* yang dibuat kita dapat mengetahui berapa besar kemenangan yang dapat diperoleh oleh min (pemain ke-2) dari setiap *state* yang dibuat. Untuk melihat *state* mana yang dapat memberikan kemenangan bagi masing-masing pemain maka kita dapat memberikan nilai atau tanda pada setiap *state.* Tandai nilai +1 untuk *state* dimana max dapat kemenangan. Dan -1 untuk min apabila min mendapat kemenangan. Dijelaskan pada gambar III.13.

Periksa mulai dari *leaf node* sampai *root*, berikan nilai untuk setiap *node* yang mengasumsikan kemenangan. Mulai dari min, berikan nilai -1 untuk *state* (1), karena *state* (1) yang menghantarkan kemenangan untuk min.

Selanjutnya maju satu tahap ke atas, giliran max yang diberikan nilai. Masih dengan mengasumsikan max akan memaximumkan langkahnya. Maka berikan *state* (2) dengan nilai +1, karena *state* (2) mengahantarkan ke *state* ( ) yang merupakan kemenangan bagi max. Dan berikan nilai +1 untuk *state* yang memiliki kondisi yang sama dengan *state* (2).

Dengan asumsi bahwa min akan mengoptimalkan langkahnya, berikan *state* (3) nilai -1 karena bagaimanapun min nantinya akan akan memilih *state* ( ). Dan berikan nilai -1 untuk *state* (1,2) karena min akan memilih *state* (1,1). Dan *state* (1,1) dengan nilai -1. Ke tahap node paling atas, berikan nilai *root* nya dengan nilai -1, karena nilai dari *subtree* sebelumnya adalah -1.



Gambar III.13 proses pemberian nilai pada *tree*

Adapun pseudocode algoritma minimaxnya:

```
Minimax (StatusSaya, Kedalaman, Pemain)
IF (Kedalaman==Max)
       RETURN static (StatusSaya, Pemain)
Bangkitkan successor S[1..n]
IF (Pemain==Saya)
       RETURN max  of Minimax (S[i], kedalaman+1, Lawan)
ELSE
 RETURN min  of Minimax (S[i], kedalaman+1, Saya)
```

III.1.4 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional adalah sebuah langkah dimana seorang pembangun perangkat lunak menganalisis sumber daya yang akan menggunakan perangkat lunak yang dibangun atau untuk menentukan spesifikasi kebutuhan sistem.

Pada analisis kebutuhan sistem nonfungsional ini dijelaskan analisis perangkat keras (*hardware),* perangkat lunak (*software),* dan pengguna (*user*) sebagai bahan analisis kekurangan dan kebutuhan yang harus dipenuhi dalam perancangan sistem yang akan diterapkan.

1. Analisis kebutuhan perangkat keras *(hardware)*

Adapun spesifikasi minimum perangkat keras yang dibutuhkan untuk menjalankan aplikasi ini, antara lain:

a. Processor intel pentium 4
b. Memori minimal 128 MB
c. Monitor resolusi 1024X768 pixel
d. Mouse

2. Analisis kebutuhan perangkat lunak (*software*)

Kebutuhan perangkat lunak yang digunakan untuk membangun aplikasi ini antara lain :

a. OS: Window
b. Bahasa pemrograman: Delphi 7.0

3. Analisis pengguna

Pemain (*Player*) untuk memainkan permainan nim. Umumnya permainan ini dapat dimainkan oleh siapapun dengan batas usia di atas 12 tahun. Khususnya

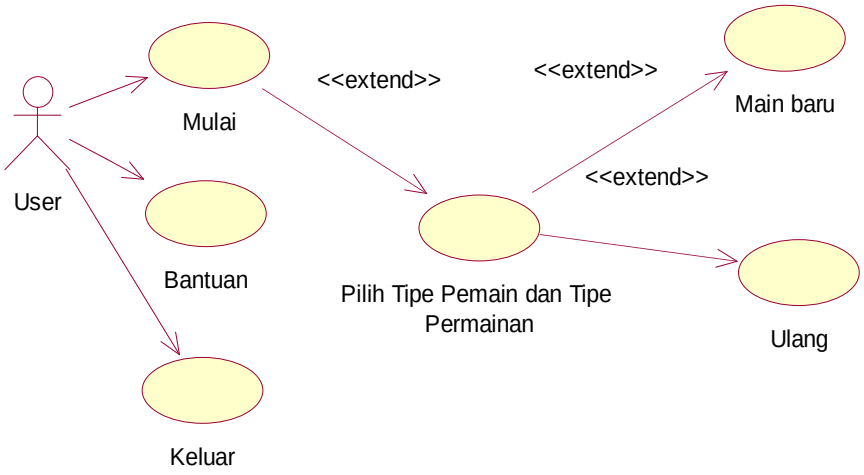
pengguna yang mengerti dan memahami komputer sehingga dapat menggunakan aplikasi yang dibangun.

III.1.5 Analisis Kebutuhan Fungsional

Adalah sumber daya analisis untuk menentukan entitas-entitas baik entitas internal maupun eksternal, data yang mengalir, serta prosedur-prosedur yang bisa dilakukan oleh masing-masing entitas.

III.1.5.1 *Use Case Diagram* dan Skenario

Use case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. *Use case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem itu digunakan. *Use case diagram* untuk aplikasi permainan nim ini dapat dilihat pada Gambar III.13. pada gambar terdapat beberapa *use case*, di antaranya: *use case* mulai, *use case* pilih tipe pemain dan tipe permainan, *use case* main baru, *use case* ulang, *use case* bantuan, dan *use case* keluar.



Gambar III.14 *Use Case Diagram*

Sedangkan skenario menjelaskan setiap proses yang terjadi dari awal hingga akhir dari sebuah *use case*. Berikut adalah skenario dari *use case diagram* yang dibuat :

1. Skenario *Use Case* Mulai

Tabel III.2 Skenario Use Case Mulai

Identifikasi Masalah	
Nama	*Use Case* Mulai
Tujuan	Memulai permainan
Deskripsi	*User* memilih menu mulai untuk memulai permainan
Aktor	*User*
Usecase Yang Berkaitan	*Use Case* Pilih tipe pemain dan tipe permainan
Skenario Utama	
Kondisi Awal	Aplikasi sudah dibuka, menampilkan menu utama
Aksi Aktor	**Reaksi Sistem**
1. Klik tombol mulai	2. Menampilkan form pengaturan
3. Menentukan tipe pemain dan tipe permainan	
4. Klik ok	5. Menampilkan permainan
Kondisi Akhir	Form Pengaturan ditampilkan

2. Skenario *Use Case* Pilih Tipe Pemain dan Tipe Permainan

Tabel III.3 Skenario Use Case Pilih Tipe Pemain dan Tipe Permainan

Identifikasi Masalah	
Nama	*Use Case* Pilih Tipe Pemain dan Tipe Permainan
Tujuan	Memilih tipe pemain dan tipe permainan
Deskripsi	Merupakan proses untuk me-*setting* tipe permainan dan tipe pemain agar permainan dapat dijalnkan sesuai yang diinginkan
Aktor	*User*
Usecase Yang Berkaitan	*Use Case* Main baru dan *Use Case* Ulang
Skenario Utama	
Kondisi Awal	Form pengaturan sudah dibuka
Aksi Aktor	**Reaksi Sistem**
1. pilih tipe pemain	
2. pilih tipe permainan	
2. Klik tombol ok	3. Menutup form pengaturan
	4. Menampilkan form utama permainan
Kondisi Akhir	Form utama permainan ditampikan

3. Skenario *Use Case* Main baru

Tabel III.4 Skenario Use Case Main Baru

Identifikasi Masalah	
Nama	*Use Case* Main Baru
Tujuan	Mengganti permainan dengan permainan baru
Deskripsi	Proses ini dilkukan jika *user* ingin mengganti permainan dengan permainan baru
Aktor	*User*
Usecase Yang Berkaitan	*Use Case* ppilih tipe pemain dan tipe permainan
Skenario Utama	
Kondisi Awal	Aplikasi sudah dibuka, form utama ditampilkan
Aksi Aktor	**Reaksi Sistem**
1. Klik tombol main baru 3. Pilih ya	2. Muncul pesan konfirmasi ‘‘ganti permainan?’’ 4. Mengaktifkan permainan 5.Setting permainan sesuai pengaturan tipe permainan yang sudah dipilih 6.Permainan ditampilkan
Skenario Alternatif (Jika Gagal)	
Aksi Aktor	**Reaksi Sistem**
1. Pilih tidak	2. Melanjutkan permainan yang sedang dimainkan
Kondisi Akhir	Mengganti dengan permainan baru

4. Skenario *Use Case* Ulang

Tabel III.5 Skenario Use Case Ulang

Identifikasi Masalah	
Nama	*Use Case* Ulang
Tujuan	Mengulangi permainan yang sudah/ sedang dimainkan dari mulai awal
Deskripsi	Merupakan proses untuk mengulangi permainan yang sudah/ sedang dimainkan dari mulai awal
Aktor	*User*
Usecase Yang Berkaitan	*Use Case* Pilih tipe pemain dan tipe permainan
Skenario Utama	
Kondisi Awal	Aplikasi sudah dibuka, form utama sudah ditampilkan
Aksi Aktor	**Reaksi Sistem**
1. Klik tombol Ulang	2. Menampilkan pesan konfirmasi ‘Ulangi permainan?’
3. Pilih ya	4. Mengulangi permainan (*restoregame*) 5. Menampilkan permainan
Skenario Alternatif (Jika Gagal)	
Aksi Aktor	**Reaksi Sistem**
1. Pilih tidak	2. Melanjutkan permainan terakhir yang sedang dimainkan
Kondisi Akhir	Mengulangi permainan

5. Skenario *Use Case* Bantuan

Tabel III.6 Skenario *Use Case* Bantuan

Identifikasi Masalah	
Nama	*Use Case* Bantuan
Tujuan	Melihat form petunjuk permainan dan Tentang aplikasi
Deskripsi	Merupakan proses untuk Melihat form petunjuk permainan dan Tentang aplikasi
Aktor	*User*
Usecase Yang Berkaitan	-
Skenario Utama	
Kondisi Awal	Aplikasi sudah dibuka, form menu utama ditampilkan
Aksi Aktor	**Reaksi Sistem**
1. Pilih *RadioButton* dalam bantuan	
2. Pilih petunjuk	3. Menampilkan form petunjuk
4. Pilih Tentang aplikasi	5. menampilkan form tentang aplikasi
Kondisi Akhir	Menampilkan form petunjuk dan tentang aplikasi

6. Skenario *Use Case* Keluar

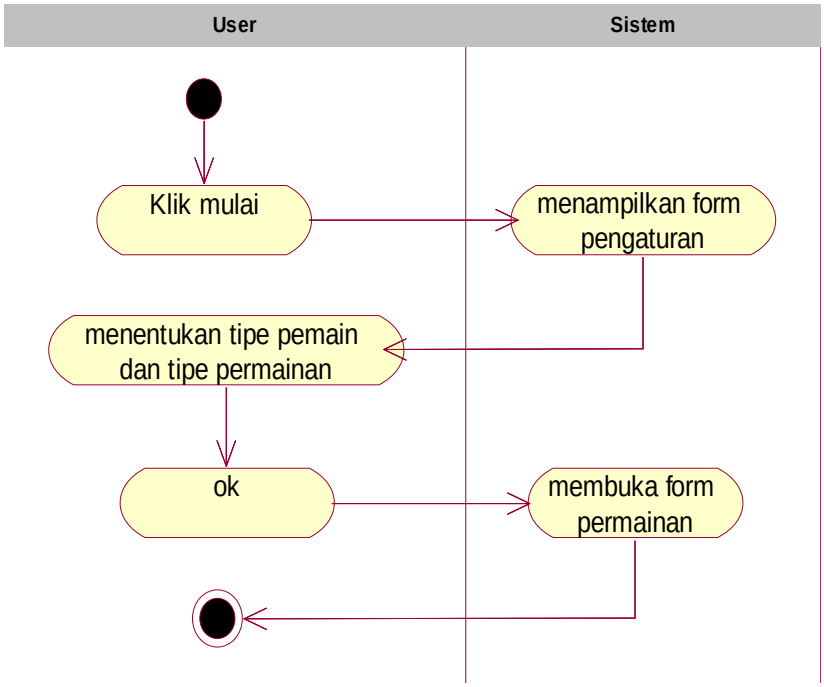
Tabel III.7 Skenario Use Case Keluar

Identifikasi Masalah	
Nama	*Use Case* Keluar
Tujuan	Menutup aplikasi
Deskripsi	Setelah proses ini dilakukan maka sistem akan menutup aplikasinya
Aktor	*User*
Usecase Yang Berkaitan	-
Skenario Utama	
Kondisi Awal	Aplikasi sudah dibuka
Aksi Aktor	**Reaksi Sistem**
1. Klik tombol Keluar	2. Menampilkan pesan konfirmasi ‘keluar aplikasi?’
3. Pilih ya	4. Menutup aplikasi
Skenario Alternatif (Jika Gagal)	
Aksi Aktor	**Reaksi Sistem**
1. Pilih tidak	2. Kembali ke form menu utama
Kondisi Akhir	Aplikasi dihentikan

III..1.5.2 Activity Diagram

Activity diagram merupakan teknik untuk menggunakan logika prosedural, proses bisnis, dan jalur kerja. Dari skenario yang dibuat, berikut akan digambarkan *activity diagram*nya.

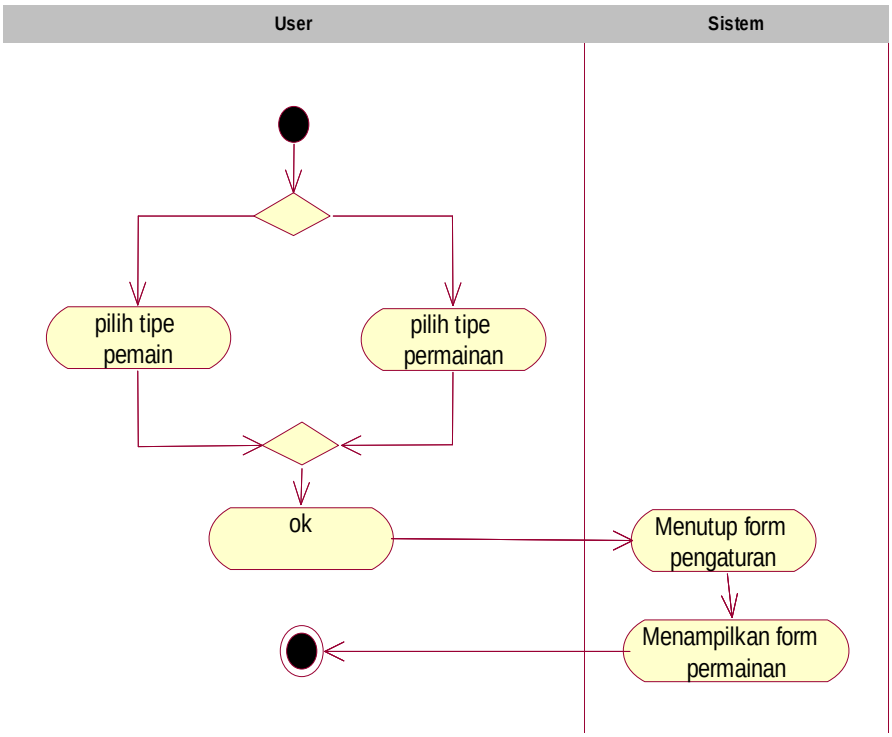
1. *Activity Diagram* Mulai



Gambar III.15 *Activity Diagram* Mulai

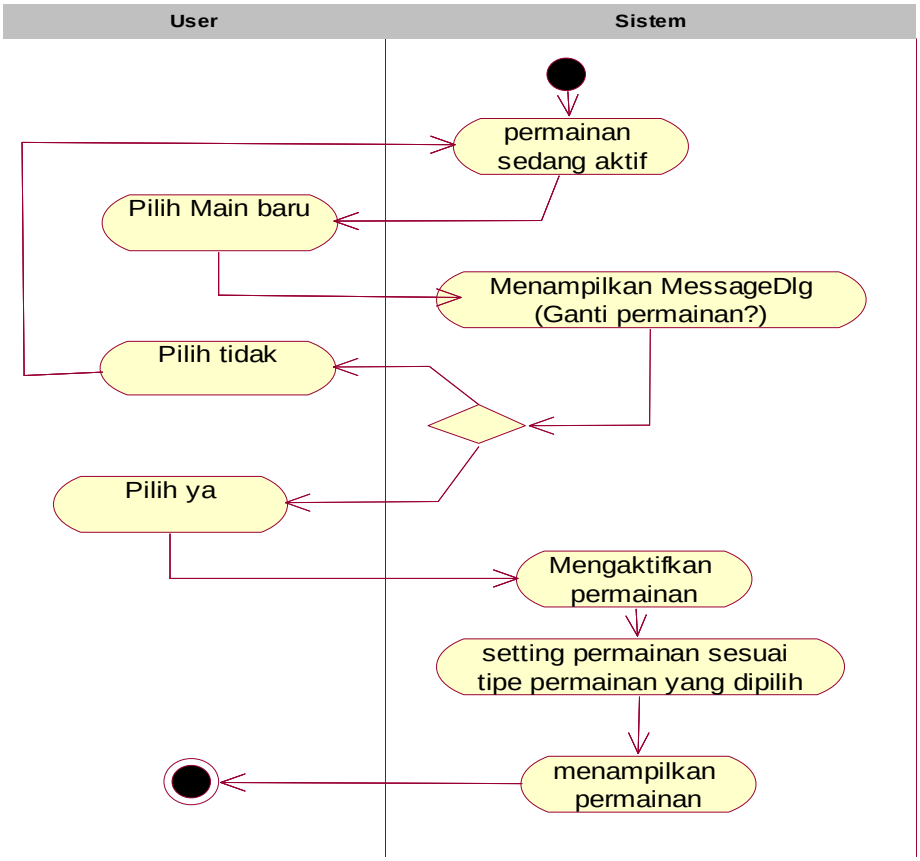
2. *Activity Diagram* Pilih Tipe Pemain dan Tipe Permainan

Dalam *activity diagram* dijelaskan bahwa *user* dapat memilih tipe pemain dan tipe permainan kemudian sistem akanmenutup form pengaturannya dan langsung menampilkan permainan



Gambar III.16 *Activity Diagram* Pilih Tipe Pemain dan Tipe Permainan

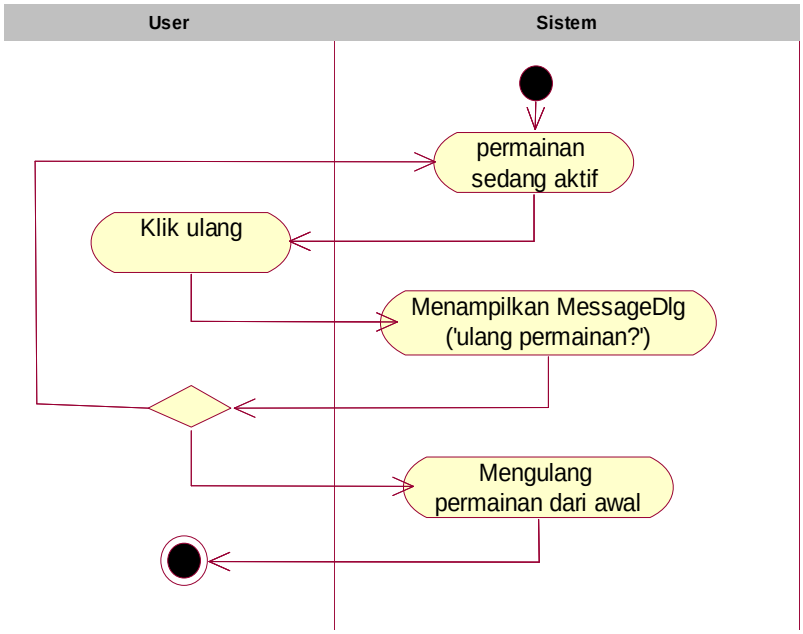
3. *Activity Diagram* Main Baru



Gambar III.17 *Activity Diagram* Main Baru

4. *Activity Diagram* Ulang

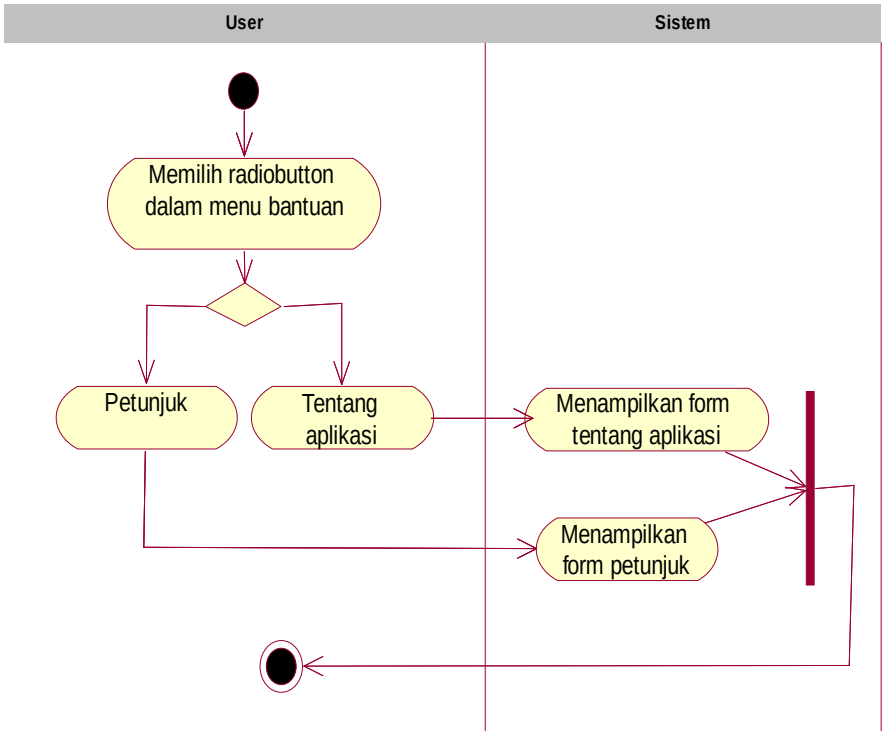
Aplikasi dalam keadaan sedang aktif, *user* kemudian pilih ulang, sistem akan menampilkan pesan dialog ‘ulang permainan?’. Jika iya, sistem akan mereset permainan terakhir yang dimainkan.



Gambar III.18 *Activity Diagram* Ulang

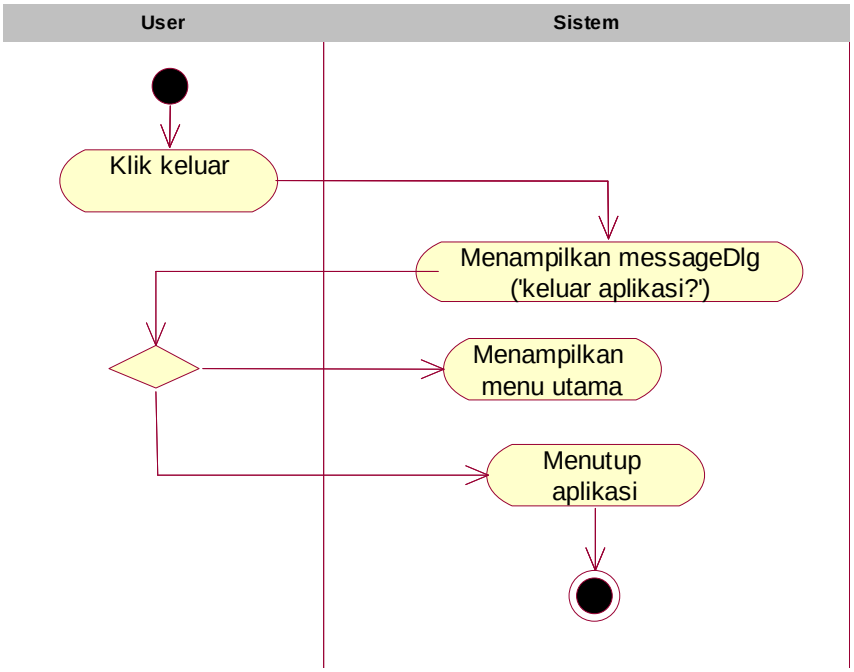
5. *Activity Diagram* Bantuan

User dapat memilih menu petunjuk atau tentang aplikasi, kemudian sistem akan menampilkan form petunjuk jika user memilih petunjuk. Dan sistem akan menampilkan tentang aplikasi jika user memilih tentang aplikasi.



Gambar III.19 *Activity Diagram* Bantuan

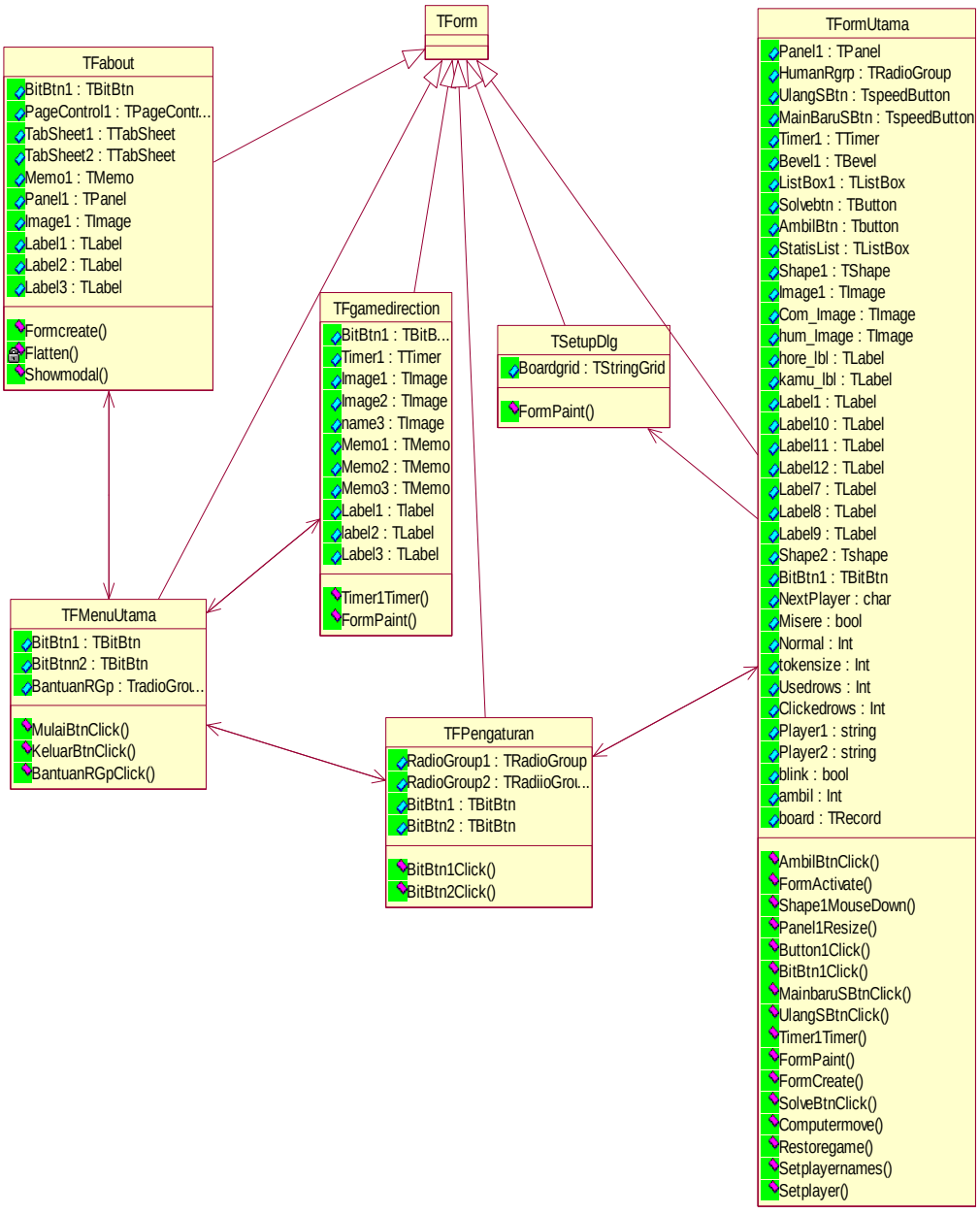
6. *Activity Diagram* Keluar



Gambar III.20 *Activity Diagram* Keluar

III.1.5.3 *Class Diagram*

Menggambarkan keterkaitan antar kelas dan mempresentasikan struktur dari sistem. Mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantara mereka.

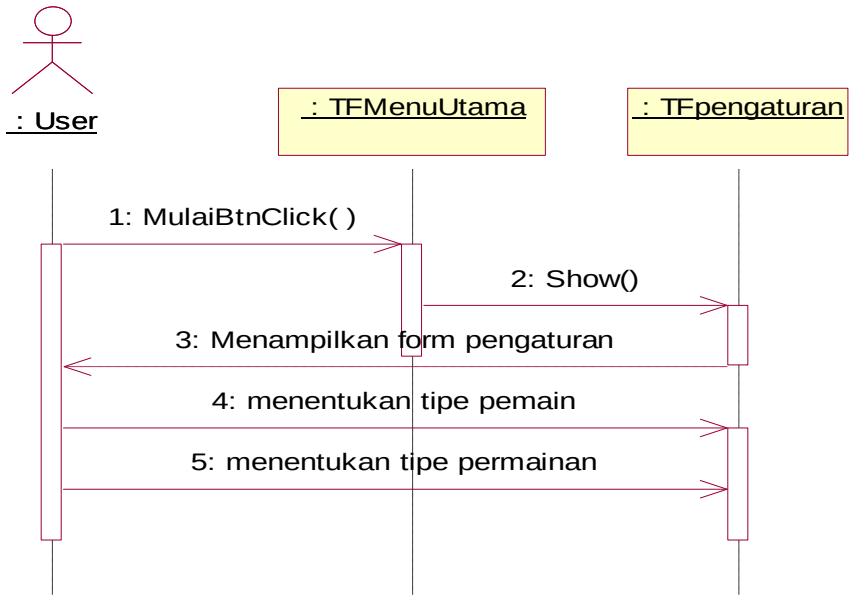


Gambar III.21 *Class Diagram*

III.1.5.4 Sequence Diagram

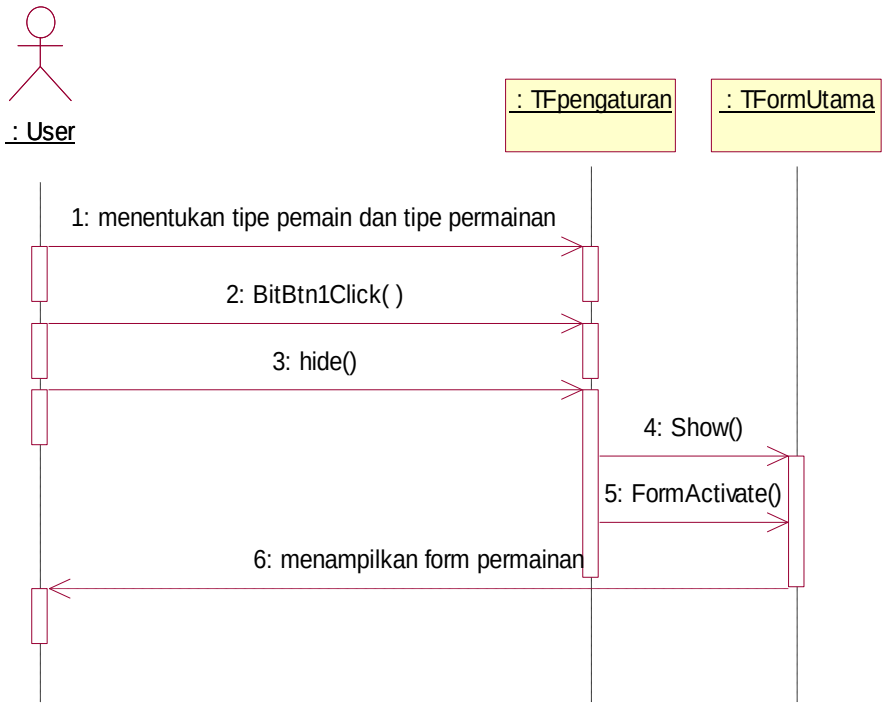
Merupakan diagram untuk menggambarkan perilaku sistem terhadap suatu interaksi yang dilakukan pada sistem tersebut. Secara khusus, menjabarkan behavior skenario tunggal. Diagram tersebut menunjukkan sejumlah objek dan pesan-pesan yang melewati objek-objek ini di dalam *use case*.

1. *Sequence Diagram* Mulai



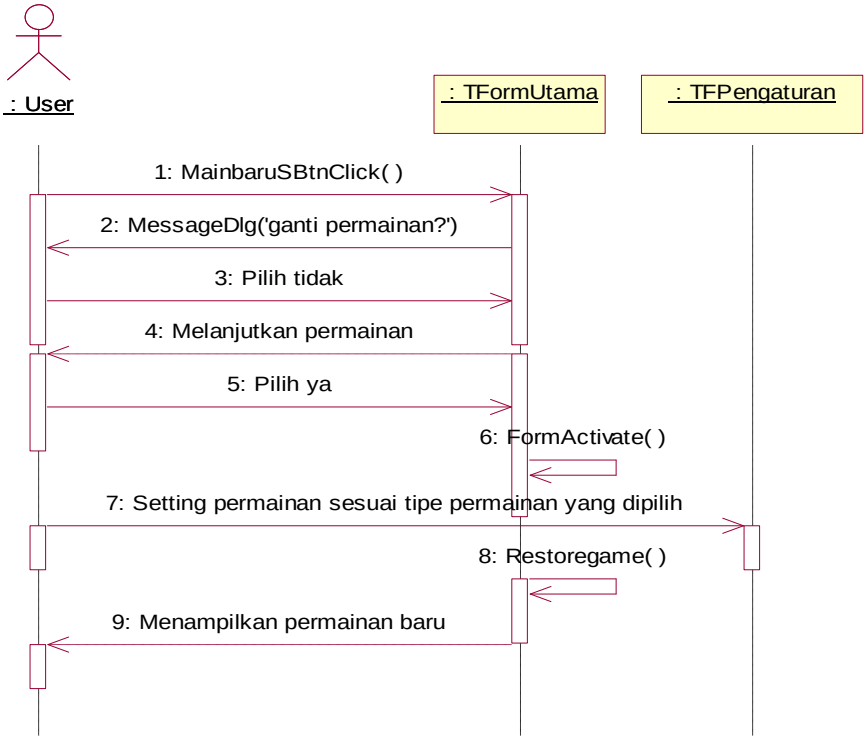
Gambar III.22 *Sequence Diagram* Mulai

2. *Sequence Diagram* Pilih Tipe Permainan dan Tipe Pemain



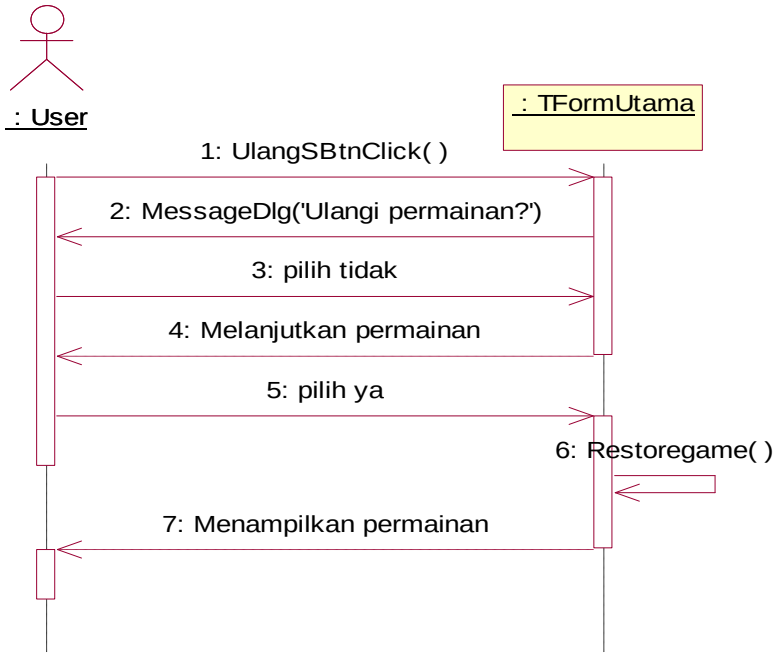
Gambar III.23 *Sequence Diagram* Pilih Tipe Permainan dan Tipe Pemain

3. *Sequence Diagram* Main Baru



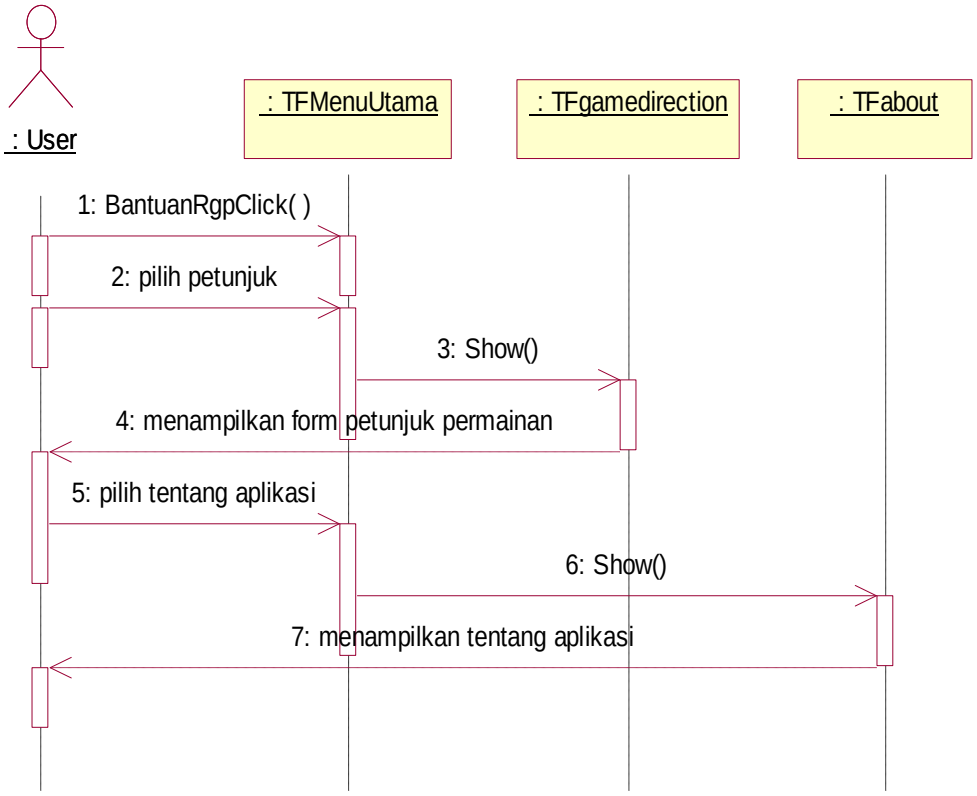
Gambar III.24 *Sequence Diagram* Main Baru

4. *Sequence Diagram* Ulang



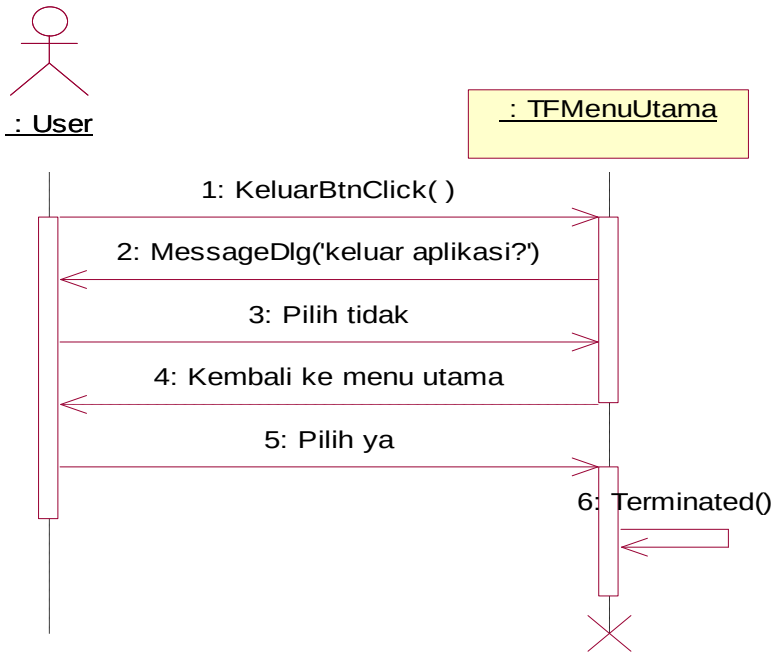
Gambar III.25 *Sequence Diagram* Ulang

5. *Sequence Diagram* Bantuan



Gambar III.26 *Sequence Diagram* Bantuan

6. *Sequence Diagram* Keluar



Gambar III.27 *Sequence Diagram* Keluar

III.2 Perancangan Sistem

Perancangan sistem merupakan metodologi pengembangan suatu perangkat lunak yang dilakukan setelah melalui tahapan analisis. Perancangan bertujuan untuk memberikan gambaran secara terperinci dan merupakan tahap lanjutan dari analisis sistem. Dimana dalam hal ini digambarkan rancangan sistem yang akan dibangun sebelum dilakukan pengodean ke dalam suatu bahasa pemrograman. Desain umum yang akan diaplikasikan bertujuan untuk

memberikan gambaran secara umum kepada penggunaan tentang sistem yang akan dibangun.

III.2.1 Perancangan Komponen Permainan

Sebelum perancangan antarmuka secara lengkap, maka terlebih dahulu harus diketahui komponen-komponen dasar dari antarmuka yang akan dirancang. Secara umum, komponen-komponen yang akan ditampilkan dalam antarmuka aplikasi yang akan dibangun adalah:

1. Pemain

Dalam permainan nim terdapat dua pemain yang akan memainkan permainan. Salah satu pemain akan bertindak sebagai *Opponent* (lawan) yang dimainkan oleh pemain manusia dan pemain lainnya adalah komputer.

2. Pengaturan Permainan

Pengaturan yang disediakan dalam permainan yaitu pilih tipe permainan dan pilih tipe pemain. Untuk pengaturan tipe permainan, ada dua pilihan, *user* dapat memilih permainan normal/biasa atau permainan misere. Sedangkan tipe pemainnya, *user* dapat memilihi apakah komputer yang akan bergerak dahulu atau *human* yang bergerak dahulu.

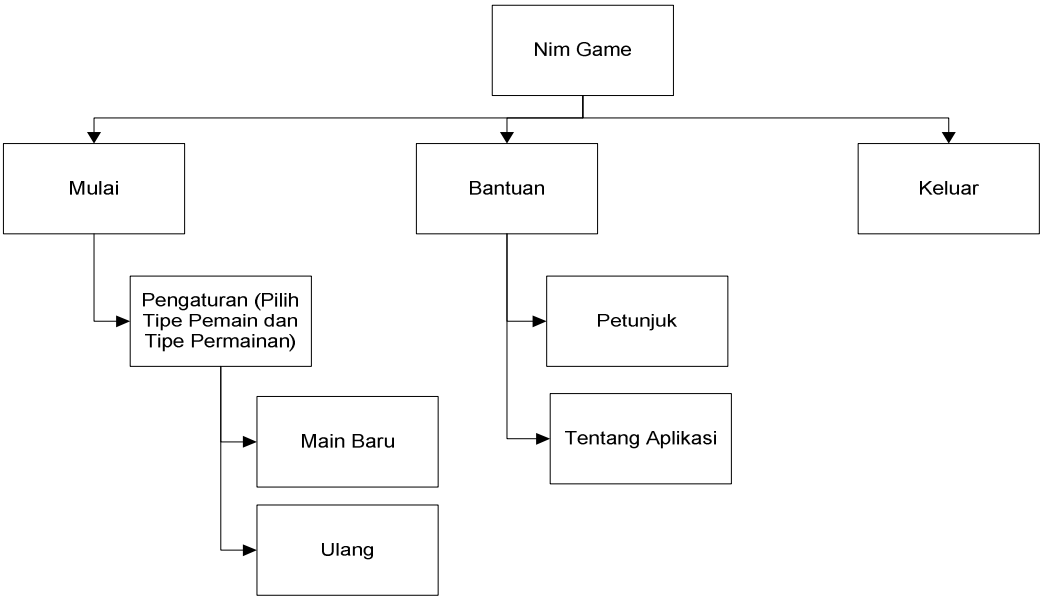
3. Cara Bermain

Adapun cara bermain yang akan diterapkan hasil dari penelitian mengenai permainan nim. Maka dalam bermain, *user* dapat mengambil beberapa buah objek pada setiap tumpukan (baris). Dengan aturan *user* tidak dapat mengambil beberapa objek sekaligus dalam dua tumpukan (baris). *User* dapat langsung bermain setelah memilih tipe pemain dan tipe permainan dalam pengaturan yang sudah disediakan. Kemudian akan disediakan rekomendasi dari komputer langkah mana yang dapat diambil pemain manusia.

4. Permainan Usai

Permainan yang akan dirancang hanya untuk sekali bermain. Apabila *user* selesai bermain maka permainan langsung berakhir. Permainan usai diidentifikasikan dengan tidak adanya jumlah objek yang tersisa.

III.2.2 Perancangan Struktur Menu

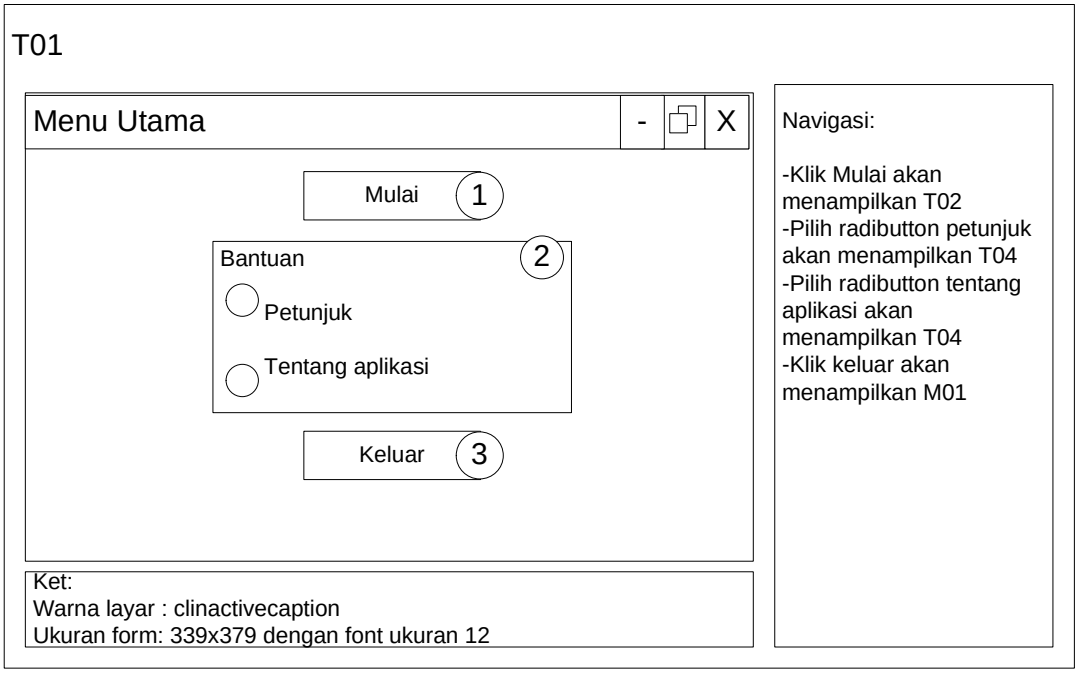


Gambar III.28 Perancangan Struktur Menu

III.2.3 Perancangan Antarmuka

Perancangan antarmuka mendeskripsikan rencana tampilan dari setiap Form yang akan digunakan pada tampilan sistem informasi sebenarnya.

1. Perancangan Form Menu Utama

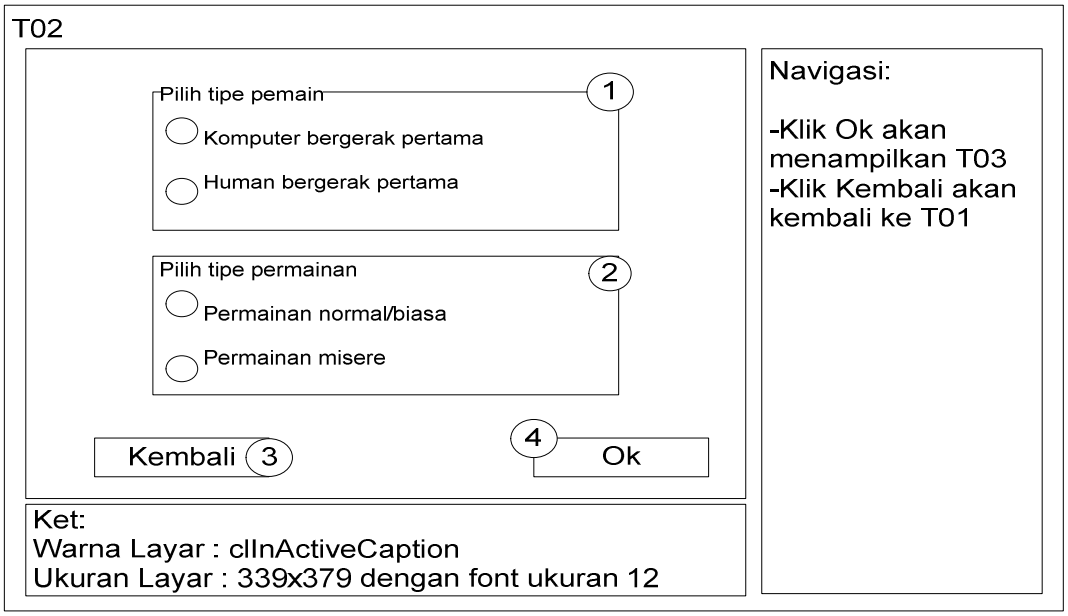


Gambar III.29 Tampilan Form Menu Utama

Tabel III.8 Komponen dalam Form Menu Utama

no	Komponen	*Properties*	Keterangan
1	*BitBtn1*	Nama: MulaiBtn	Untuk memulai permainan baru
2	*RadioButtonGroup1*	Nama: RadioButton1	Untuk melihat petunjuk dan tentang aplikasi
3	*BitBtn2*	Nama: KeluarBtn	Untuk keluar dari aplikasi

2. Perancangan Form Pengaturan



Gambar III.30 Tampilan Form Pengaturan

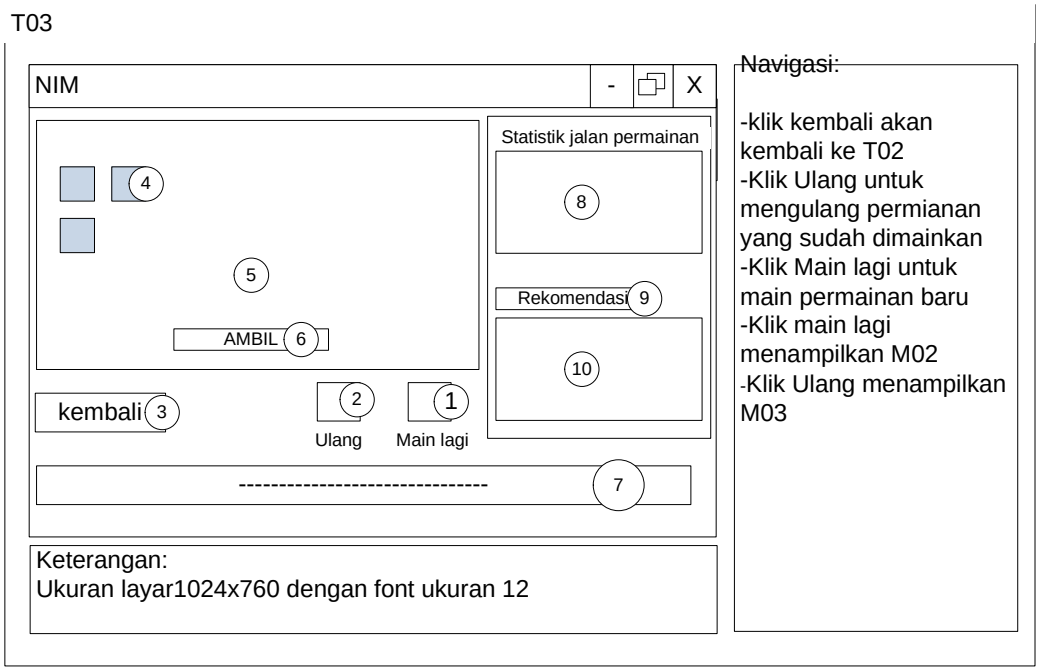
Tabel III.9 Komponen dalam Form Pengaturan

no	Komponen	*Properties*	Keterangan
1	*RadioGroup1*	Nama: *RadioGroup1* Caption : Pilih tipe pemain Items: 1. Computer pertama 2. Human pertama	Untuk menentukan siapa yang akan bergerak terlebih dahulu
2	*RadioGroup2*	Nama: *RadioGroup2* Caption : Pilih tipe permainan	Untuk menentukan tipe permainan yang ingin dimainkan

		Items: 1. Permainan Normal/biasa 2. Permainan *misere*	
3	*BitBtn2*	Nama : *BitBtn2* Caption : Kembali	Untuk kembali ke form menu utama
4	*BitBtn1*	Nama : *BitBtn1* Caption : Ok	Untuk melanjutkan ke form utama

3. Perancangan Form Utama

Form Utama ini difungsikan untuk menampilkan papan permainannya.

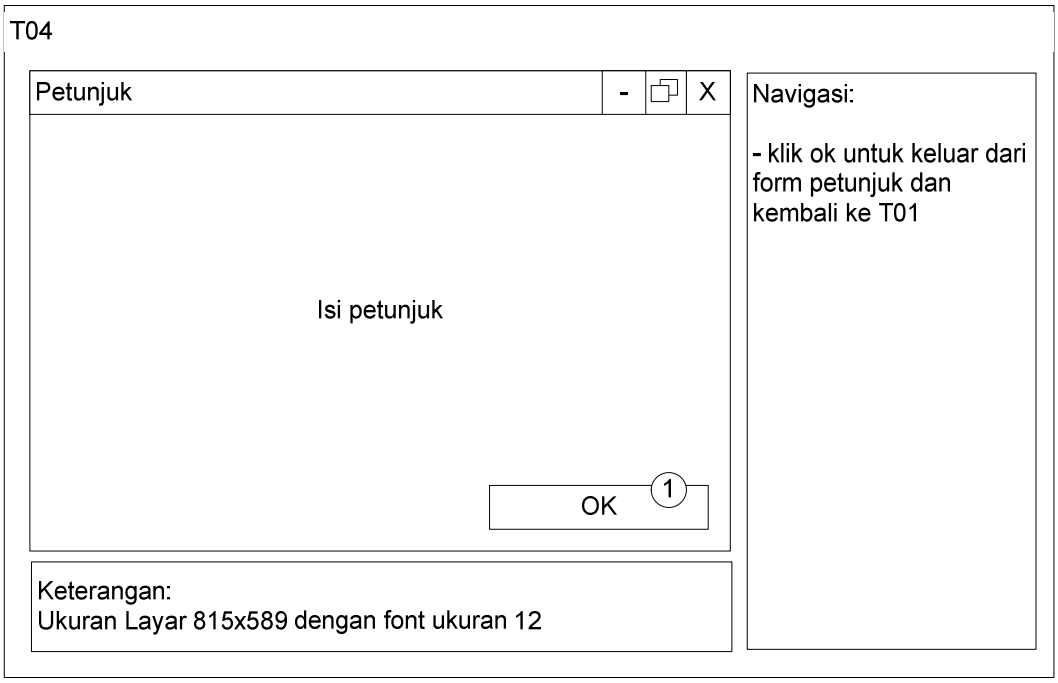


Gambar III.31 Tampilan Form Utama

Tabel III.10 Komponen Form Utama

no	Komponen	*Properties*	Keterangan
1	*SpeedButton1*	Nama: permainanbaruBtn	Untuk memulai permainan baru
2	*SpeedButton2*	Nama: ulangBtn	Untuk mengulang permainan
3	*BitBtn1*	Nama: *BitBtn1* Caption: Kembali	Untuk kembali ke form pengaturan
4	*Shape*	Nama: *shape*	Objek
5	*Panel1*	Nama: Panel1	Area permainan
6	*Button*	Nama: ambilBtn	Proses untuk menagmbil objek
7	*Bevel1*	Nama: *Bevel1*	Berisi keterangan
8	*ListBox1*	Nama: *StatisList*	Infornasi statistik jalannya permainan, giliran siapa yang bermain dan berapa jumlah objek yang diambil
9	*Button*	Nama: RekomendBtn	Proses untuk menampilkan informasi langkah yang dapatdilakukan human yang diberikan oleh komputer
10	*ListBox2*	Nama: ListBox1	Menampilkan informasi yang diberikan oleh komputer

4. Perancangan Form Petunjuk

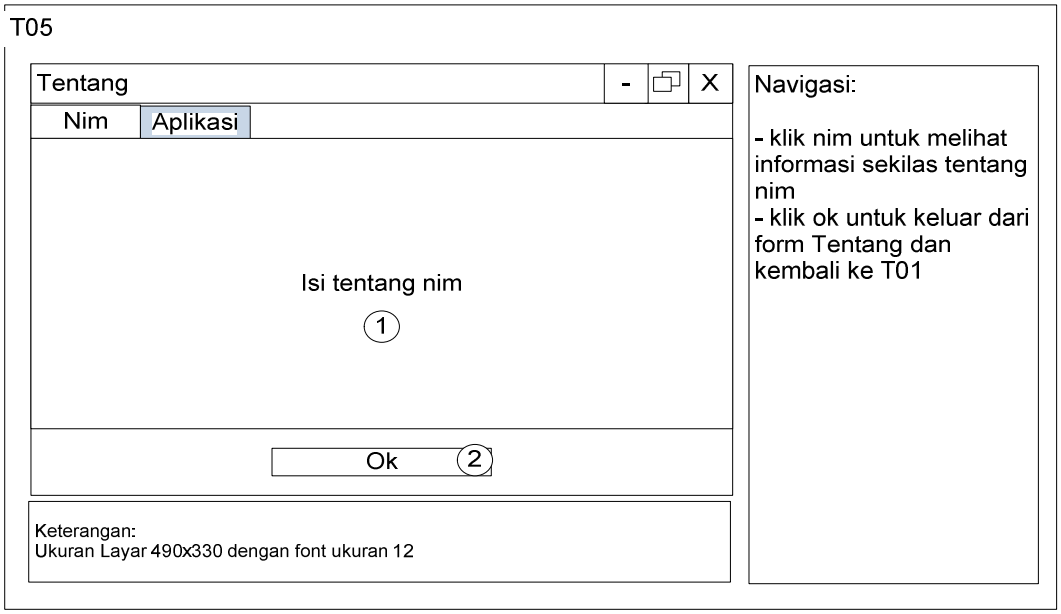


Gambar III.32 Tampilan Form Petunjuk

Tabel III.11 Komponen Form Petunjuk

No	Komponen	*Properties*	Keterangan
1	*BitButton1*	Nama: OkBtn	Untuk keluar dari form Petunjuk

5. Perancangan Form Tentang Aplikasi



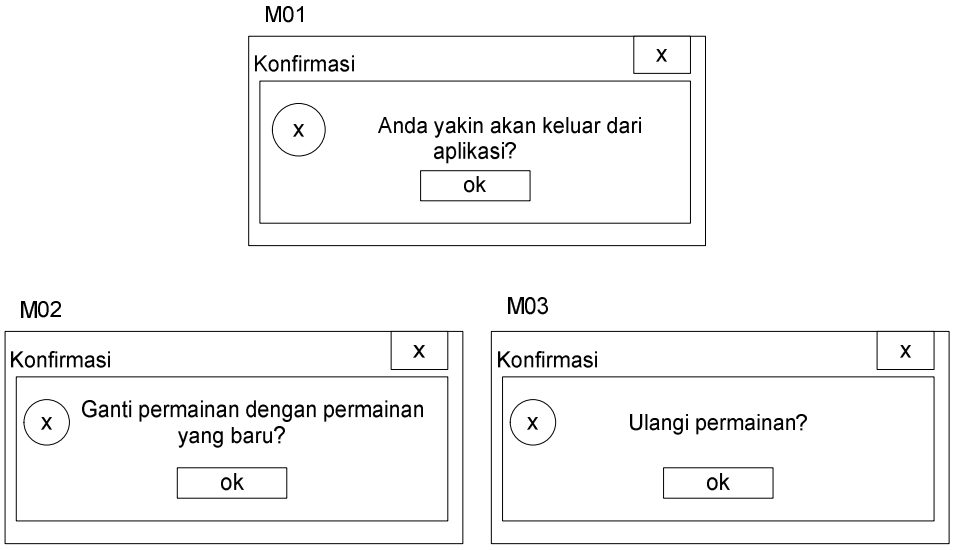
Gambar III.33 Tampilan Form Tentang Aplikasi

Tabel III.12 Komponen Form Tentang Aplikasi

No	Komponen	*Properties*	Keterangan
1	*Panel1*	Nama: Panel1	Untuk menampilkan isi dari tentang aplikasi
2	*BitButton1*	Nama: OkBtn	Untuk keluar dari form tentang

III.5.4.1 Perancangan Pesan Konfirmasi

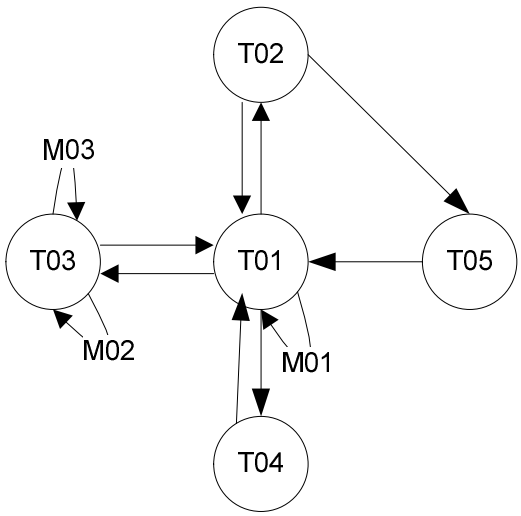
Berikut gambar perancangan pesan konfirmasi. Ada tiga *MessageDlg* yang dibuat, yaitu: pesan untuk keluar dari aplikasi, pesan untuk ganti permainan, dan pesan untuk mengulang permainan.



Gambar III.34 Perancangan Pesan Konfirmasi

III.2.4 Jaringan Semantik

Perancangan jaringan semantik menggambarkan alur informasi dari setiap antarmuka aplikasi yang dijelaskan.



Gambar III.35 Jaringan Semantik

Keterangan:

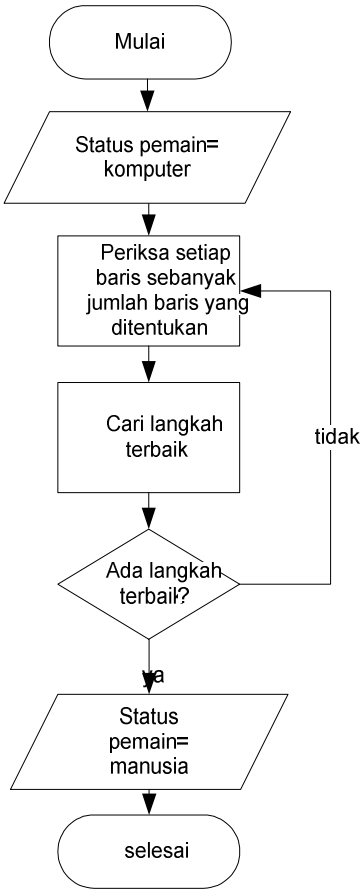
1. T01 : form menu utama
2. T02 : form pengaturan
3. T03 : form utama
4. T04 : form petunjuk
5. T05 : form tentang aplikasi
6. M01 : pesan konfirmasi keluar aplikasi
7. M02 : pesan konfirmasi ganti permainan
8. M03 : pesan konfirmasi ulang permainan

III.2.5 Perancangan Prosedural

Bagan alir sistem *(flowchart system)* merupakan bagan-bagan yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu

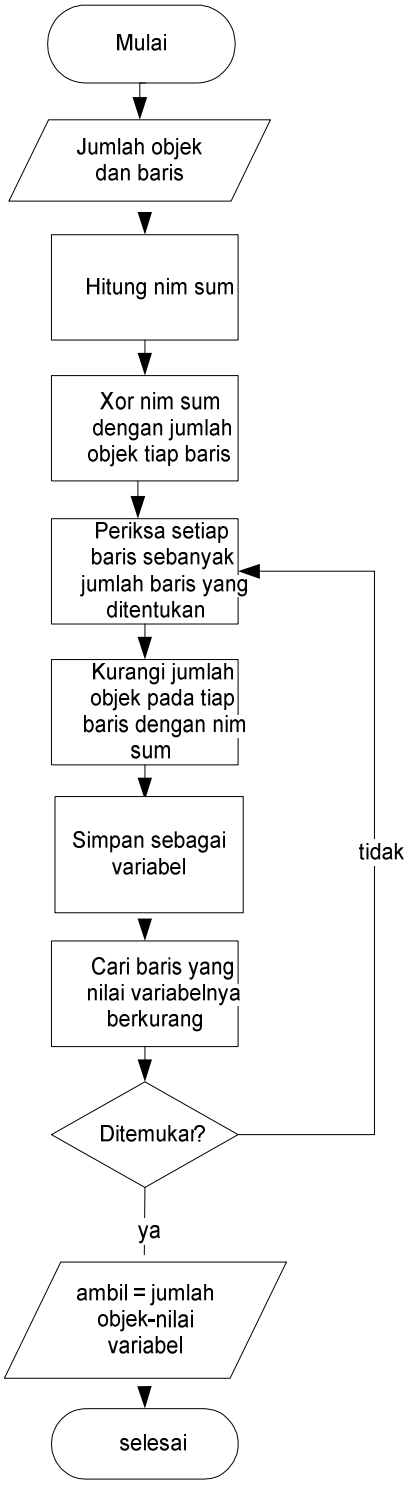
masalah dan merupakan cara penyajian dari suatu algoritma, sehingga dapat menjelaskan urutan prosedur-prosedur yang ada dalam sistem.

1. Bagan Alir Pergantian Pemain



Gambar III.36 Bagan alir perancangan pergantian pemain

2. Bagan Alir Perancangan Proses Pencarian



Gambar III.37 bagan alir perancangan proses pencarian