Rancang Bangun Perangkat Lunak Untuk Permainan Othello Multiplayer

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika

oleh:

Nama : Irawan Tunas Nugroho
No. Mahasiswa : 00 523 205

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA

2007

LEMBAR PENGESAHAN PEMBIMBING

Rancang Bangun Perangkat Lunak Untuk Permainan Othello Multiplayer

TUGAS AKHIR

oleh:

Nama : Irawan Tunas Nugroho

No. Mahasiswa : 00 523 205

Yogyakarta, 24 September 2007

Pembimbing,

Yudi Prayudi, S.Si., M.Kom.

LEMBAR PENGESAHAN PENGUJI

Rancang Bangun Perangkat Lunak Untuk Permainan Othello Multiplayer

TUGAS AKHIR

oleh:

Nama : Irawan Tunas Nugroho

No. Mahasiswa : 00523205

Telah Dipertahankan di Depan Sidang Penguji sebagai Salah Satu Syarat

untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika

Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 24 September 2007

Tim Penguji	
Yudi Prayudi, S.Si., M.Kom. Ketua	
Taufik Hidayat, ST., MCS Anggota	

Mengetahui,
Ketua Jurusan Teknik Informatika
Universitas Islam Indonesia

UNIVERSITAS ISLAM INDONESIA ★ YOGYAKARTA ★ FAKULTAS TEKNOLOGI INDUSTRI

Yudi Prayudi, S.Si.,M.Kom

PERSEMBAHAN

Karya Sederhana ini kupersembahkan untuk .....

Ayahandaku (Sidik Pramono) dan Ibundaku (Endang Sri Wahjanti) tercinta, yang telah memberikan limpahan kasih sayang dan bimbingan yang tiada akhir. Serta selalu memberikan yang terbaik untukkuTerima kasih atas segala jerih payah dan pengorbananmu.

Kakakku tersayang (Arlina Kusuma Wardhani) I Love U Sis'.

Saudaraku (Andik Yulianto) You will always gonna be My Man...

Untuk seseorang yang diciptakan ALLAH SWT hanya untukku,

Kapan kau hadir lagi untukku ? Kapan kau basuh jiwa ini dengan kasihmu?...Kehadiranmu akan selalu aku tunggu...

Sahabat-sahabatku (Informatika 2000, GPK's Squad, Wong_Edan's Crew, KKN SL-29, Kontrakan Pusung II's, Sphinx & Islamic Center's Family dan seluruh sahabatku di dunia) yang selalu mendukung, menemani dan menjadi sumber inspirasiku, kalianlah warna dalam hidupku...

.... karena mereka semualah yang telah memberikan dukungan dan kasih sayang kepadaku selama aku menyelesaikan tugas akhir. WITHOUT YOU'ALL, I'M NEVER GONNA MAKE IT ON MY OWN...

Juga untuk lilin hidupku dan penjaga hatiku yang terus menerangi dan mendamaikan jiwaku dengan cahaya--Nya yang SUCI...

MOTTO

"Orang-orang Yang Beriman dan Hati Mereka Menjadi Tentram Dengan Mengingat Allah, Ingatlah Hanya Dengan Mengingat Allah Hati Menjadi Tentram"

(QS : Ar-Ra'd : 28)

"Sesungguhnya setelah kesulitan itu ada kemudahan."

[QS : ASY SYARH : 6]

"Adalah lebih baik menjadi lilin yang menyinari secara sederhana tetapi lama, dibanding menjadi kembang api yang menyinari secara indah tetapi hanya sesaat"

(Agung Hadi Mulya)

"Didalam kegagalan, yang ada hanyalah BELUM BERHASIL bukanlah TIDAK BERHASIL, MAJU TERUS... karena akan lebih mengecewakan jika kita harus MENYERAH "

(Sms yang dikirimkan Andik)

"Jika kau dalam kesulitan, janganlah kau katakan pada Tuhanmu bahwa kau menghadapi masalah yang besar, tetapi katakanlah pada masalahmu bahwa engkau memiliki Tuhan yang BESAR"

"SAY WHAT YOU'VE GOT TO SAY and DO WHAT YOU'VE GOT TO DO"

KATA PENGANTAR

لسلم عليكم ورحمة الله وبركاته

Assalamu'alaikum Warohmatullahi Wabarokatuh

Alhamdulillahi Rabbil'alamiin segala puji bagi Allah Tuhan semesta alam. Berkat rahmat dan hidayah-Nya sehingga Tugas Akhir yang berjudul "RANCANG BANGUN PERANGKAT LUNAK UNTUK PERMAINAN OTHELLO MULTIPLAYER" dapat terselesaikan. Shalawat dan salam semoga senantiasa tercurah kepada Muhammad SAW sebagai uswah dan inspirator perjuangan penyusun, beserta keluarga, sahabat dan umatnya hingga hari kiamat.

Tugas Akhir ini diajukan sebagai salah satu syarat untuk memperoleh gelar sarjana Teknik Informatika Fakultas Teknologi Industri Univeraitas Islam Indonesia. Dalam menyelesaikan Tugas Akhir ini penyusun mendapat masukan dari berbagai pihak yang sangat bermanfaat bagi penyusun.

Penyusun dalam kesempatan ini menghaturkan terima kasih kepada :

1. Ibunda dan Ayahanda tercinta, atas curahan bahasa kasih sayang yang tak terbalaskan.
2. Bapak Fathul Wahid, ST., M.SC selaku Dekan Fakultas Teknologi Industri.
3. Bapak Yudi Prayudi, S.Si., M.Kom selaku Ketua Jurusan Teknik Informatika dan selaku Dosen Pembimbing.
4. Kakakku tersayang Arlina, dengan dorongan semangat yang tak ada habisnya.
5. Teman - teman angkatan 2000 semuanya.
6. Agnes, dengan do'a dan perhatiannya.

7. Kawan – kawan Islamic Centre Bojonegoro, dengan dukungan dan do'anya.
8. Semua Pihak yang memberikan kontribusi dalam penyelesaian Tugas Akhir ini.

Dalam penyelesaian Tugas Akhir ini penyusun menyadari bahwa masih banyak terdapat kesalahan dan kekurangannya, untuk itu penulis mengharapkan kritik dan saran yang membangun agar bisa berguna untuk masa mendatang.

Akhir kata, Semoga Tugas Akhir ini dapat bermanfaat bagi semua untuk diamalkan dan menjadi barokah dunia sampai akhirat.

Amiin.

والسلم عليكم ورحمة الله وبركاته

Wassalamu'alaikum Warohmatullahi Wabarokatuh

Yogyakarta, 24 September 2007

Penyusun

DAFTAR ISI

HALAMAN JUDUL i

LEMBAR PENGESAHAN PEMBIMBING ii

LEMBAR PENGESAHAN PENGUJI iii

HALAMAN PERSEMBAHAN iv

HALAMAN MOTTO v

KATA PENGANTAR vi

DAFTAR ISI viii

DAFTAR TABEL xii

DAFTAR GAMBAR xiii

ABSTRAKSI xxii

BAB I PENDAHULUAN 1

1.1 Latar Belakang Masalah 1

1.2 Rumusan Masalah 2

1.3 Batasan Masalah 2

1.4 Tujuan Penelitian 3

1.5 Manfaat Penelitian 3

1.6 Metodologi Penelitian 3

1.7 Sistematika Penulisan 4

BAB II LANDASAN TEORI 8

2.1 *Game* 8

2.1.1 Sejarah *Game* 8

2.1.2 Definisi *Game* 10

2.2 Othello 10

2.2.1 Sejarah 10

2.2.2 Definisi 11

2.2.3 Peraturan Permainan 11

2.3 Teori Dalam Pembuatan Program Permainan 13

2.4 *Unified Modeling Language* (UML) 16

2.4.1 Definisi 16

2.4.2 Konsep Dasar UML 16

2.5 *Open GL* 27

2.5.1 Definisi 27

2.5.2 Spesifikasi 28

2.5.3 Desain 28

2.5.4 Open GL Buffer 31

2.6 TCP/IP 33

2.6.1 Definisi 33

2.6.2 Protocol Operation 34

BAB III ANALISIS KEBUTUHAN PERANGKAT LUNAK 36

3.1 Metode Analisis 36

3.2 Hasil Analisis Kebutuhan 36
3.2.1 Kebutuhan Masukan 36
3.2.2 Kebutuhan Keluaran 37
3.2.3 Fungsionalitas yang dikehendaki 37
3.3 Analisis Kebutuhan Antarmuka 37
3.3.1 Analisis kebutuhan perangkat lunak 38
3.3.2 Analisis kebutuhan perangkat keras 38
BAB IV PERANCANGAN PERANGKAT LUNAK 39
4.1 Metode Perancangan 39
4.2 Hasil Perancangan 39
4.2.1 Diagram *Use Case* 39
4.2.2 Diagram Aktivitas 40
4.2.3 Diagram *Class* 41
4.2.4 Diagram *Sequence* 43
4.3 Aturan Bermain 49
4.4 Perancangan Antar muka 50
BAB V IMPLEMENTASI PERANGKAT LUNAK 55
5.1 Batasan Implementasi. 55
5.1.1 Asumsi yang dipakai 55
5.1.2 Lingkungan Pengembangan 56
5.1.3 Perangkat lunak yang digunakan 56
5.1.4 Perangkat keras yang digunakan 56

5.2 Implementasi 57

5.2.1 Implementasi Interface 57

5.2.2 implementasi Procedural 63

BAB VI ANALISIS KINERJA PERANGKAT LUNAK 72

6.1 Analisis Kesesuaian 72

6.1.1 Analisis perhitungan data pada koordinat 73

6.2 Pengujian program 74

6.2.1 Pengujian pada pemain 77

6.3 Hasil Analisis 77

6.3.1 Perbandingan dengan Othello yang ada 78

BAB VII KESIMPULAN DAN SARAN 80

7.1 Kesimpulan 80

7.2 Saran 80

DAFTAR PUSTAKA

DAFTAR TABEL

Tabel 2.1 Konsep Dasar UML 16

Tabel 6.1 Pengujian pada beberapa orang pemain..... 77

Tabel 6.2 Perbandingan dengan game Othello yang sudah ada 78

DAFTAR GAMBAR

Gambar 2.1 Posisi buah di awal permainan 11

Gambar 2.2 Langkah yang valid untuk pemain pertama 12

Gambar 2.3 Langkah yang valid untuk pemain kedua 13

Gambar 2.4 Contoh Diagram Use Case 18

Gambar 2.5 Contoh Diagram Class 19

Gambar 2.6 Contoh Diagram Statechart 20

Gambar 2.7 Contoh Diagram Aktivitas 22

Gambar 2.8 Contoh Diagram Sequence 24

Gambar 2.9 Contoh Diagram Kolaborasi 25

Gambar 2.10Contoh diagram Komponen 26

Gambar 2.11Contoh Diagram Deployment 27

Gambar 2.12Proses Diagram Pipa 30

Gambar 2.13TCP *State Diagram* 34

Gambar 4.1 Diagram *Use Case* 40

Gambar 4.2 Diagram Aktivitas 41

Gambar 4.3 Diagram *Class* 43

Gambar 4.4 Diagram *Sequence Network Game* 44

Gambar 4.5 Diagram *Sequence Refresh Online* 45

Gambar 4.6 Diagram *Sequence Start* 46

Gambar 4.7 Diagram *Sequence Help* 47

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Pada era globalisasi seperti saat ini, komputerisasi hampir menyentuh segala aspek kehidupan manusia. Mulai dari bidang industri, kesehatan, pendidikan, perbankan dan bahkan untuk media hiburan yaitu berupa permainan (*game*). Kemajuan di bidang teknologi erat kaitannya dengan teknologi informasi berbasis komputer ini sangatlah cepat dan keberadaannya sangatlah dibutuhkan, sehingga ada pepatah yang berbunyi "Orang yang menguasai informasi akan menguasai dunia". Hal ini tidaklah berlebihan karena tuntutan kegiatan bisnis yang semakin kompleks dan teknologi komputer telah mencapai kemampuan yang menakjubkan sehingga informasi menjadi suatu sumber daya yang sangat dibutuhkan.

Game telah lama dimainkan oleh manusia dan ini merupakan hal yang wajar bagi semua kultur budaya di dunia. Manusia menggunakan *game* sebagai sarana untuk refreshing dan menghilangkan ketegangan setelah melakukan rutinitas, dan pada *game* tertentu juga bisa digunakan untuk belajar.

Othello atau yang dikenal dengan nama lain *Reversi* merupakan salah satu permainan strategi dengan menggunakan papan. Walaupun permainan ini tidak seterkenal catur , namun permainan ini sangat digemari orang-orang di seluruh dunia. Terlebih lagi *game* ini dapat dimainkan oleh dua orang.

Seiring dengan perkembangan teknologi khususnya di bidang komputer, tampaknya permainan ini akan lebih menarik bila diimplementasikan dalam bentuk perangkat lunak permainan komputer, dan dapat dimainkan oleh dua orang (*Multiplayer*) dengan menggunakan jaringan komputer.

1.2 Rumusan Masalah

Dari uraian diatas dapat ditarik suatu rumusan masalah yaitu : Bagaimana merancang, membangun dan mengimplementasikan permainan *Othello Multiplayer* dalam suatu program/aplikasi perangkat lunak, yang sekaligus dapat menjadi salah satu alternatif permainan yang sudah ada dan sekaligus dapat dijadikan perbandingan dengan permainan yang sudah ada sebelumnya.

1.3 Batasan Masalah

Implementasi yang dilakukan pada penelitian tugas akhir ini mempunyai batasan masalah sebagai berikut :

1. Ukuran atau jumlah papan tempat untuk meletakkan buah dibatasi dengan 8 x 8 (baris x kolom).
2. Permainan dua orang (pemain vs pemain) dapat dilakukan dalam dua komputer dengan menggunakan jaringan.
3. Pemain melangkah harus melompati buah pemain lawan dan kotak tujuan adalah kotak kosong.

4. Score dihitung berdasarkan buah yang dimiliki pemain.
5. Permainan akan berakhir bila kedua pemain tidak dapat melangkah atau papan permainan telah penuh.
6. Pada permainan Othello Network Game, permainan akan selesai apabila papan permainan telah penuh dan apabila salah satu pemain berhasil menutup semua bidak dari pemain lawan dan papan permainan akan secara otomatis akan tertutup.

1.4 Tujuan Penelitian

Tujuan utama penelitian tugas akhir ini adalah untuk mengimplementasikan secara nyata permainan *Othello Multiplayer* ke dalam komputer.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah agar dapat menghasilkan perangkat lunak *game Othello* yang menyenangkan.

1.6 Metodologi Penelitian

Adapun metode penelitian yang digunakan dalam penelitian ini yaitu :

a. Pengumpulan Data.

Mengumpulkan data lewat buku-buku referensi, dokumen, artikel dan catatan lain yang *relevan* dengan permasalahan yang dihadapi.

b. Analisis Kebutuhan dan Perancangan

Pada tahap ini dilakukan proses analisis terhadap berbagai kebutuhan yang mungkin diperlukan oleh sistem yang akan dibangun dan dilanjutkan dengan proses perancangan aplikasi perangkat lunak.

c. Implementasi

Proses penerapan desain dengan menggunakan alat bantu yang paling efisien dan efektif untuk mencari pemecahan masalah demi mencapai tujuan yang diinginkan. Implementasi merupakan tahap penerapan semua algoritma dan prosedur yang telah disusun dalam langkah perancangan sistem.

e. Analisis Kerja

Tahap pengujian terhadap perangkat lunak yang telah diimplementasikan.

1.7 Sistematika Penulisan

BAB I PENDAHULUAN

Pada Bab ini pertama diawali dengan penjelasan mengenai latar belakang masalah, kemudian dilanjutkan dengan rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelititan, metodologi penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI

Pada Bab ini membahas landasan teori yang dapat membantu para pembaca dalam memahami implementasi yang akan dilakukan.

Landasan teori tersebut mencakup tentang *game*, *Othello*, aturan permainan *Othello*, *Unified Modelling Language* (UML), *Open GL* dan protokol *TCP/IP*.

BAB III ANALISIS KEBUTUHAN PERANGKAT LUNAK

Bab ini berisi mengenai :

Metode Analisis

Berisi mengenai metode-metode yang dipakai pada analisis kebutuhan perangkat lunak.

Hasil Analisis

Berisi mengenai hasil analisis, yaitu pemilihan kebutuhan-kebutuhan dalam pembuatan perangkat lunak yang meliputi : fungsi-fungsi yang dibutuhkan, kinerja atau tujuan yang harus dicapai serta pemilihan kebutuhan dalam pembuatan antarmuka (*interface*).

BAB IV PERANCANGAN PERANGKAT LUNAK

Bab ini berisi mengenai :

Metode Perancangan

Berisi mengenai metode-metode yang dipakai dalam perancangan perangkat lunak.

Hasil Perancangan

Berisi mengenai hasil perancangan perangkat lunak, hasil perancangan merupakan terjemahan kebutuhan perankat lunak yang meliputi struktur

data, arsitektur perangkat lunak, prosedur-prosedur dan antarmuka (*interface*).

BAB V IMPLEMENTASI PERANGKAT LUNAK

Batasan Implementasi

Berisi mengenai batasan implementasi perangkat lunak yang meliputi : asumsi-asumsi yang dipakai dan batasan lain yang dibuat dan ditemui selama pengembangan perangkat lunak.

Implementasi

Berisi mengenai dokumen implementasi perangkat lunak yang meliputi : prosedur-prosedur dalam bahasa pemrograman yang dipilih serta antarmuka (*interface*). Khusus untuk prosedur-prosedur disertai keterangan untuk setiap proses yang ada.

BAB VI ANALISIS KINERJA PERANGKAT LUNAK

Berisi mengenai dokumentasi hasil pengujian terhadap perangkat lunak yang dibandingkan kebenaran dan kesesuaiannya dengan kebutuhan perangkat lunak untuk kemudian dianalisa.

BAB VII PENUTUP

Kesimpulan

Berisi kesimpulan-keimpulan dari proses pengembangan perangkat lunak, baik pada tahap analisis kebutuhan perangkat lunak, perancangan implementasi dan terutama pada analisis kinerja perangkat lunak.

Saran

Berisi mengenai saran yang perlu diperhatikan berdasar keterbatasan-keterbatasan yang ditemukan dan asumsi yang dibuat selama melaksanakan tugas akhir.

BAB II

LANDASAN TEORI

2.1 Game

2.1.1 Sejarah Game

Game telah dimainkan oleh manusia selama beribu-ribu tahun lamanya dan ini merupakan hal yang wajar bagi semua kultur budaya di dunia. Sepanjang sejarah di bumi, manusia telah menggunakan tongkat untuk menggambar *game board* yang simpel di permukaan tanah, kemudian membuat beberapa aturan untuk menyusun batu atau obyek lain untuk kemudian dimainkan. Sekitar 5000 tahun yang lalu, manusia mulai membuat *game board* yang lebih permanen, mereka menggunakan kayu atau lempengan tanah liat. *Senet* adalah salah satu *game* pertama di dunia yang berasal dari Mesir. Seperti *game* awal lainnya, *senet* mempunyai makna religius. Orang-orang Mesir percaya bahwa gambar-gambar yang berada pada kotak papan permainan melambangkan perjalanan hidup jiwa manusia setelah meninggal.

Beberapa *game board* yang tertua dikembangkan dari metode ramalan, atau menceritakan masa depan. Ada juga *game* yang memerlukan kecermatan yang tinggi seperti *game* strategi. Lebih dari 3000 tahun yang lalu, di China telah dikembangkan sebuah *game* strategi yang berasal dari metode ramalan, pemain melemparkan/menempatkan lempengan putih dan hitam pada garis/simbol dalam papan permainan, di mana garis atau symbol tersebut mempunyai arti tersendiri.

Banyak *game* modern yang dikembangkan setelah abad masehi, *Game* tersebar ke seluruh penjuru bumi kemudian orang-orang mulai mencoba membuat aturan baru, menciptakan variant atau mengubah *game* yang asli sesuai dengan kehendaknya. Sebagai contoh adalah *mancala*, sebuah *game* matematik yang dimainkan oleh orang-orang Mesir, di mana kerikil, benih(biji), atau obyek lain ditaruh dalam lubang-lubang dalam sebuah papan kayu. Setelah *mancala* tersebar di Asia, Afrika dan Amerika, pemain local mulai mengembangkannya ke dalam beberapa versi yang sampai sekarang masih dimainkan. Dua contoh versi dari *mancala* adalah *sungka* (dari Philipina) dan *mweso* (dari Uganda).

Catur, *weiqi*(catur China), dan *shogi* (catur Jepang) adalah jenis *game board* yang paling banyak dimainkan di seluruh dunia. Walaupun sungguh berbeda, ketiganya dipercaya berasal dari moyang yang sama. Setelah berabad-abad lamanya, catur menyebar dari barat ke Timur Tengah dan Eropa, dengan berbagai macam perubahan pada aturan-aturannya. Catur juga menyebar ke Korea dan Jepang, sehingga menghasilkan aturan-aturan yang sangat berbeda.

Sepanjang sejarah manusia, suatu *game* akan mudah populer jika *game* tersebut mudah dimainkan dan bahan-bahan untuk membuatnya juga mudah didapat. Adanya mesin percetakan (yang ditemukan pada pertengahan tahun 1400) membuat proses penyebaran *game* menjadi lebih mudah. Namun, hal ini tidak terlalu hebat jika dibandingkan dengan revolusi industri (pada abad ke 18) yang mengakibatkan mudahnya membuat versi *game* yang baru dari berbagai bahan. Perkembangan

teknologi pada abad ke 20 seperti ditemukannya plastik dan revolusi komputer mendorong terciptanya lebih banyak *game* dengan berbagai macam variasi dan model yang dikombinasikan dengan abad-abad sebelumnya.

2.1.2 Definisi Game

Seperti yang tercantum dalam ensiklopedia situs encarta.msn.com bahwa definisi *game* adalah *"activities or contest governed by sets of rules"* .*Game* haruslah mempunyai aturan-aturan dan mempunyai suatu tujuan (*goal*) yaitu menang, kalah atau seri. *Game* bersifat interaktif dan melibatkan pemain, inilah yang membedakan *game* dar hiburan (*shows).*

2.2 OTHELLO

2.2.1 Sejarah Othello

Othello dikenal dengan nama Reversi merupakan salah satu permainan strategi dengan papan. Tidak ada data pasti mengenai asal usul permainan ini, tetapi kebanyakan orang menganggap Othello berasal dari China sengan nama 'Fan Mian', sementara yang lainnya beranggapan ditemukan oleh Lewis Waterman dan John W. Mollett. Sekitar tahun 1970, seseorang berkebangsaan Jepang bernama Gozo Hasegawa membuat peraturan baru, yang kini digunakan di seluruh dunia.

2.2.2 Definisi

Seperti yang tercantum dalam situs id.wikipedia.org bahwa definisi Othello (Reversi) adalah suatu permainan strategi di mana kita harus mengisi ruang-ruang kosong (berbentuk kotak hijau) dengan buah-buah dimana pemain melangkah harus melompati buah pemain lawannya dan kotak tujuan adalah kotak kosong. Langkah boleh horizontal, vertical dan silang tergantung kotak-kotak yang harus diisi berdasarkan aturan. Dalam penelitian tugas akhir ini, ruang-ruang kosong (berbentuk kotak hijau) tidak mutlak harus berwarna hijau.

2.2.3 Peraturan Permainan

Seperti yang tercantum dalam situs http://www.Othello.org.hk. Untuk memulai permainan, awal permainan dengan aturan modern adalah seperti formasi dibawah ini dengan 8x8 kotak. Untuk giliran siapa yang melangkah terlebih dahulu, ini merupakan kesepakatan pemain.

```
  1 2 3 4 5 6 7 8
+-----------------+
1|. . . . . . . .|
2|. . . . . . . .|
3|. . . . . . . .|
4|. . . o # . . .|
5|. . . # o . . .|
6|. . . . . . . .|
7|. . . . . . . .|
8|. . . . . . . .|
+-----------------+
```

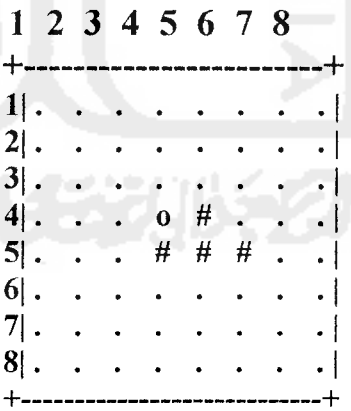
Gambar 2.1 Posisi Buah di awal Permainan

Misalkan pemain pertama (#) dan pemain kedua (o).

Aturan bermain:

1. Pemain melangkah harus dengan melompati buah pemain lawannya dan kotak tujuan adalah kotak kosong.
2. Langkah boleh horizontal, vertical dan silang.
3. Nilai masing-masing pemain dihitung berdasarkan buah yang dimiliki masing-masing pemain.
4. Bila salah satu pemain tidak dapat melangkah, maka pemain tersebut harus pass, dan pemain lawannya akan melangkah lagi.
5. Permainan akan berakhir bila kedua pemain tidak dapat melangkah lagi ataupapan permainan telah penuh.

Contoh langkah yang valid:



Gambar 2.2 Langkah yang Valid Untuk Pemain 1

Pemain pertama (#) seandainya melangkah ke 56 (baris 5 kolom 6) maka buah o milik pemain pertama akan berubah menjadi #. Maka scorenya pemain pertama (#) : 4 dan pemain (o) : 1.

Bila pemain kedua (o) melakukan langkah ke 66 (baris 6 kolom 6) maka :

```
  1 2 3 4 5 6 7 8
+-----------------+
1|. . . . . . . .|
2|. . . . . . . .|
3|. . . . . . . .|
4|. . . o # . . .|
5|. . . # o # . .|
6|. . . . . o . .|
7|. . . . . . . .|
8|. . . . . . . .|
+-----------------+
```

Gambar 2.3 Langkah yang Valid Untuk Pemain 2

Score permainan menjadi 3 : 3.

Demikian seterusnya.

2.3 Teori Dalam Pembuatan Program Permainan

Urutan pembuatan dan pengembangan program *game* adalah sebagai berikut :

1. Tentukan tipe permainan apa yang ingin dibuat.

   Penentuan ini sebagai dasar sebelum mulai bekerja sampai didapat ide yang kira-kira bagus untuk dibuat program permainan.

2. Definisikan model permainan dan tujuannya.

   Ide yang didapat itu dituangkan dalam bentuk model permainan yang dibuat. Pada tahap ini sebaiknya model permainan ditulis secara jelas sehingga jika hendak menambah cerita, karakter lain dalam permainannya atau suatu aksi baru, maka bisa dilihat atau dicek dari yang sudah ditulis sehingga permainannya tetap konsisten dan tidak membingungkan.

3. Definisikan secara jelas GAME WORLDS-nya.

   Games worlds adalah elemen-elemen utama yang terdapat dalam suatu program permainan yang terdiri dari :

   a. Game Board

      Bentuk tampilan permainan, latar belakang, dan lainnya.

   b. Instruksi untuk permainan

      Instruksi untuk pemain harus jelas supaya tidak membingungkan dan pemain dapat menentukan strategi dari permainannya.

   c. Informasi untuk pemain

      Informasi ini penting ditampilkan dalam program permainan ketika sedang berjalan, misalnya skor, waktu dan lainnya.

   d. Penghargaan

      Penghargaan memegang peranan penting dalam programpermainan karena dengan adanya penghargaan (ucapan selamat dan lainnya) setelah bermain

maka penghargaan akan merangsang pemain untuk memainkan level yang lebih tinggi lagi.

e. Variasi

Program permainan tanpa variasi yang memadai akan membuat orang cepat bosan tetapi varasi juga tidak boleh berlebihan sehingga akhirnya tidak jelas karena selalu ada kemungkinan variasi yang berlebihan akan membuat alur permainan tidak konsisten.

4. Pastikan bahwa bisa dimainkan

Permainan haruslah dapat dimainkan, karena tampilan grafik yang luar biasa cepat atau permainan yang tidak dapat diselesaikan akan membuat program permainan tidak dapat dimainkan.

5. Rancang program sebaik mungkin

Gunakan teknik pemrograman yang sesuai dalam membuatnya dan selalu buatlah supaya program mudah dimodifikasi.

6. Pengujian Program

Program yang selesai dibuat harus diuji. Pertama oleh perancangnya sendiri (alpha tes) untuk menentukan kesalahan logis yang mungkin terjadi. Kedua oleh orang lain (beta tes) untuk mengetahui kesalahan lain yang tidak terlihat oleh perancangnya sehingga dapat memberikan masukan, saran atau ide dalam pengembangan program yang sedang dibuat.

2.4 Unified Modelling Language

2.4.1 Definisi

Unified Modelling Language (UML) adalah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentsikan sistem perangkat lunak [DHA03]. Dalam hal ini UML menawarkan sebuah standar untuk merancang model sebuah sistem.

2.4.2 Konsep Dasar UML

Menurut Sri Dharwiyanti dan Romi Satria Wahono dalam artikel mereka "Pengenalan Unified Modelling Language (UML)" bahwa dari berbagai penjelasan rumit yang terdapat di dokumen dan buku-buku UML, sebenarnya konsepsi dasar UML dapat dirangkum dalam Tabel 2.1.

Tabel 2.1 Konsep Dasar UML [DHA03]

Major Area	*View*	*Diagrams*	*Main Concepts*
Structural	*Static view*	*Class Diagram*	*Class, association, generalization, dependency, realization, interface*
	Use case view	*Use case diagram*	*Use case, actor, association, extend, include, use case generalization*
	Implementation view	*Component diagram*	*Component, interface, dependency, realization*

	Deployment view		
		Deployment diagram	*Node, component, dependency, location*
Dynamic	*State machine view*	*Statechart diagram*	*State, event, transition, action*
	Activity view	*Activity diagram*	*State ,activity, completion transition, fork, join*
	Interaction view	*Sequence diagram*	*Interaction, object, message, activation*
		Collaboration diagram	*Collaboration, interaction, collaboration role, message*
Model management	*Model management view*	*Class diagram*	*Package, subsystem, model*
Extensibility	*All*	*All*	*Constraint, stereotype, tagged values*

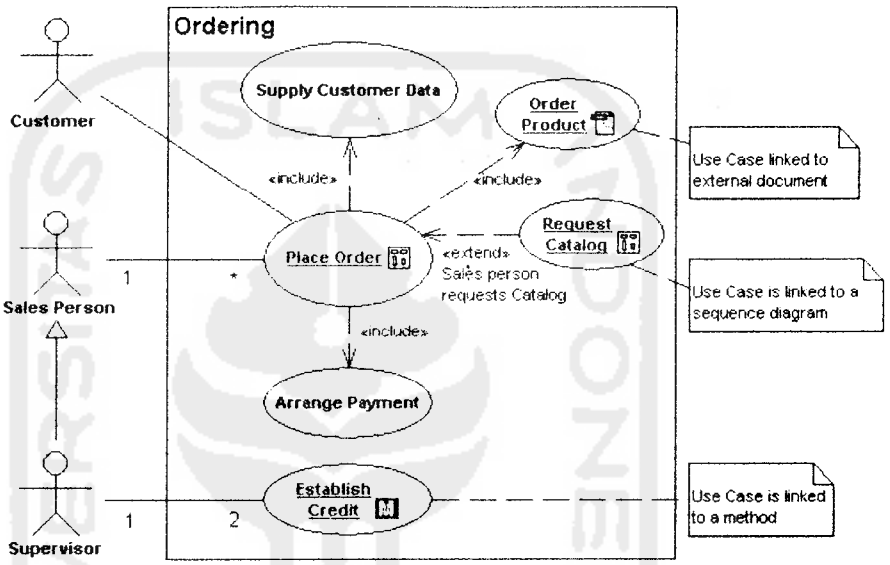
Seperti tercantum dalam tabel di atas UML mendefinisikan diagram-diagram sebagai berikut :

1. Diagram *use-case (use-case diagram)*

   Use case diagram are description of functionality of the system from the user's perspective. Use case diagram are used to show the functionality that the system will be provide and to show which user will communicate with the system in some way to use that functionality [BEN02].

   Dalam diagram *use-case* terdapat tiga aspek yang penting, yaitu : *use case, actor* dan *system boundary* atau *sub-system boundary*. *Use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem. *Use case* merepresentasikan interaksi antara *actor* dengan sistem. *Actor* adalah sebuah

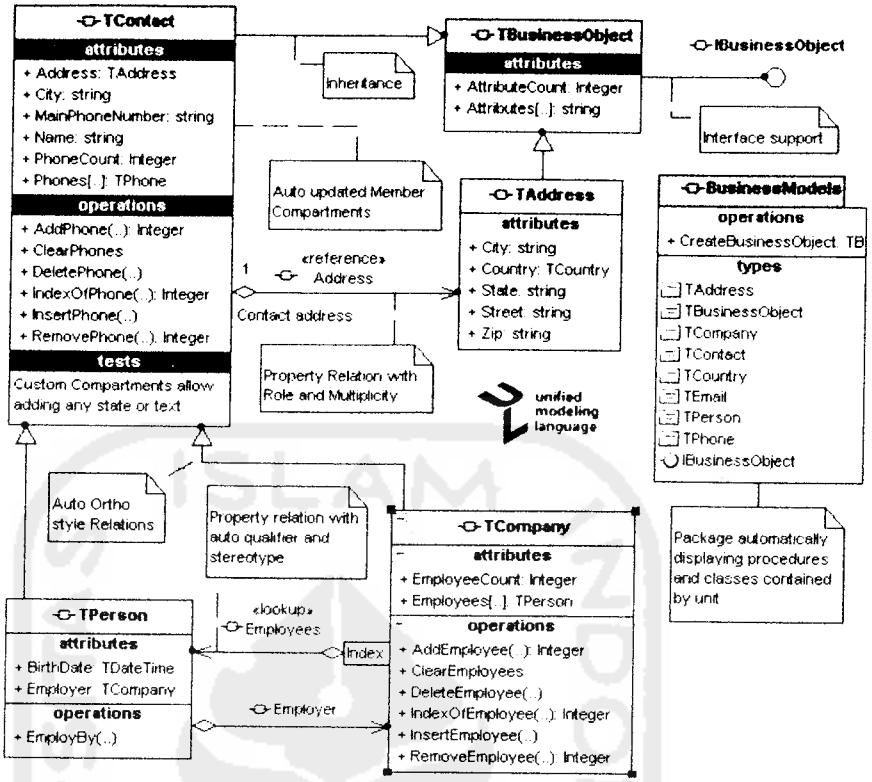
entitas berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. Sedangkan *system boundary* adalah batasan sistem. Sebuah sistem harus mempunyai batasan, jika tidak, maka dia akan gagal menyebutdirinya sebagai sistem. Contoh diagram *use-case* dapat dilihat pada Gambar 2.4.



Gambar 2.4 Contoh diagram *use- case* [MOD05]

2. Diagram *class (class diagram)*

Diagram *class* menggambarkan struktur dan deskripsi *class*, *package* dan *obyek* beserta hubungan satu sama lain seperti *containment,* pewarisan, asosiasi dan lain-lain. Contoh diagram *class* dapat dilihat pada gambar 2.5.



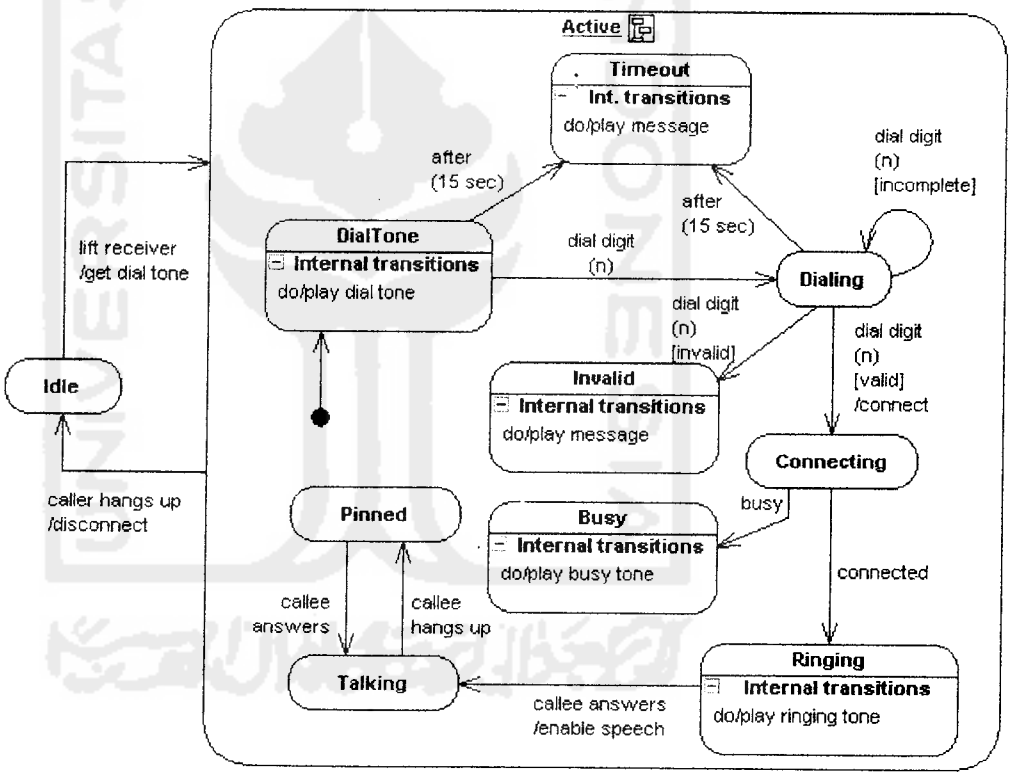
Gambar 2.5 Contoh *class diagram* [MOD05]

3. Diagram *statechart (statechart diagram)*

Diagram *statechart* menggambarkan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu obyek pada sistem sebagai akibat dari *stimuli* yang diterima. Pada umumnya diagram *statechart* menggambarkan *class* tertentu (satu *calss* dapat memiliki lebih dari satu diagram *statechart*) [DHA03].

Dalam UML, *state* digambarkan berbentuk segiempat dengan sudut membulat dan memiliki nama sesuai kondisinya saat itu. Transisi antar *state* umumnya

memiliki kondisi *guard* yang merupakan syarat terjadinya transisi yang bersangkutan, dituliskan dalam kurung siku. *Action* yang dilakukan sebagai akibat dari *event* tertentu dituliskan dengan diawali dengan garis miring [DHA03]. Titik awal dan akhir digambarkan berbentuk lingkaran berwarna penuh dan berwarna setengah. Contoh diagram *statechart* dapat dilihat pada gambar 2.6.



Gambar 2.6 Contoh diagram *statechart* [MOD05]

4. Diagram aktivitas *(activity diagram)*

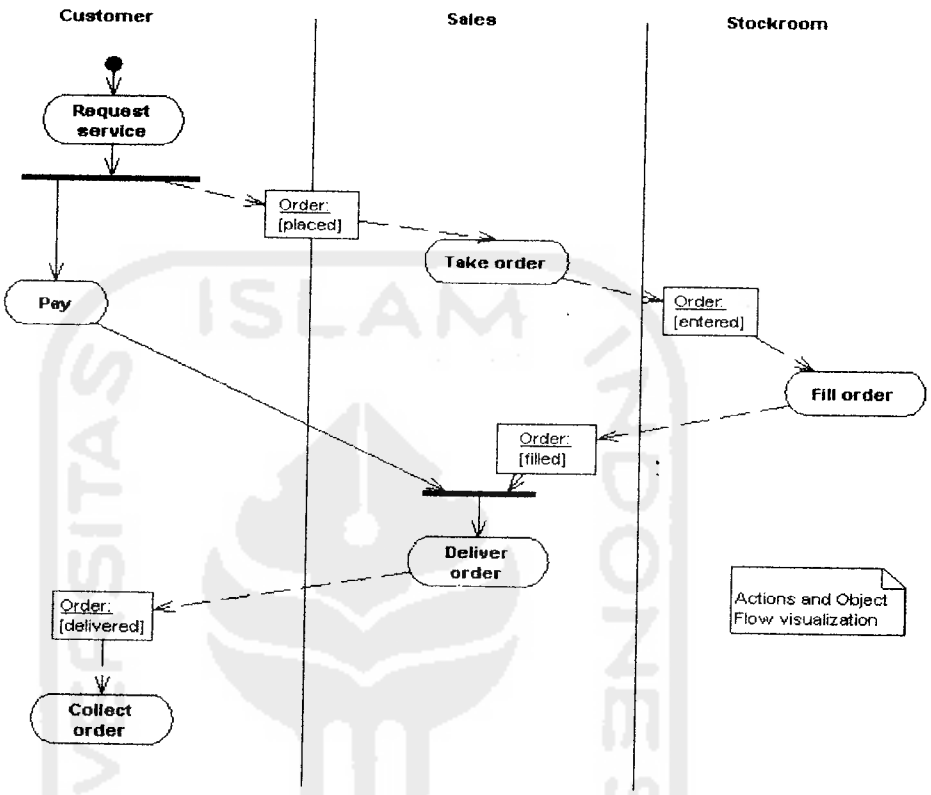
Diagram aktivitas menggambarkan berbagai alir aktivitas dalam system yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. Diagram aktivitas juga dapat menggambarkan proses parallel yang mungkin terjadi pada beberapa eksekusi [DHA03].

Diagram aktivitas merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu diagram aktivitas tidak menggambarkan *behaviour* internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dar level atas secara umum [DHA03].

Sebuah aktivitas dapat direalisasikan oleh suatu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas [DHA03].

Sama seperti *state,* standar UML menggunakan segiempat dengan sudut membulat dan menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behaviour* pada kondisi tertentu. Untuk mengilustrasikan proses-proses parallel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. Diagram aktivitas dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan obyek mana yang

bertanggung jawab untuk aktivitas tertentu. Contoh diagrm aktivitas dapat dilihat pada Gambar 2.7.



Gambar 2.7 Contoh diagram aktivitas [MOD05]

5. Diagram *sequence (sequence diagram)*

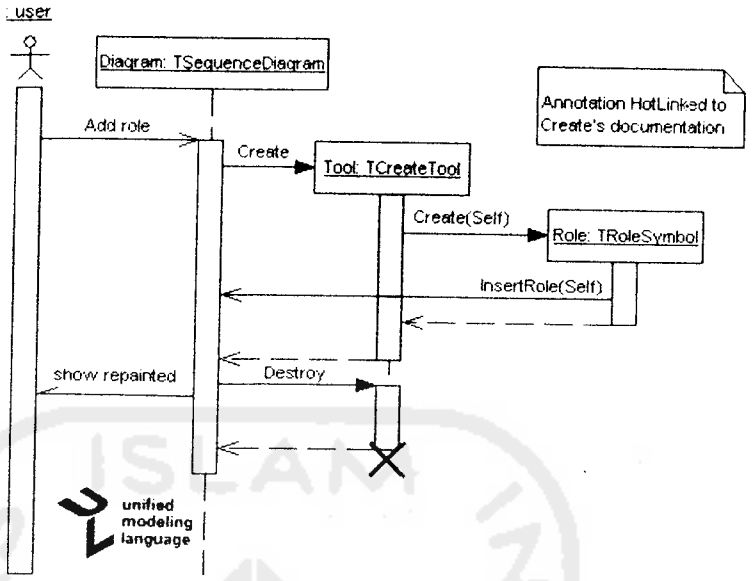
Diagram *sequence* menggambarkan interaksi antar obyek di dalam dan di sekiter sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. Diagram *sequence* terdiri antar dimensi vertikal (waktu) dan dimensi horizontal (obyek-obyek yang terkait) [DHA03].

Diagram *sequence* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan [DHA03].

Masing-masing obyek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu obyek ke obyek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation bar* menunjukkan lamanya eksekusi sebuah proses. Biasanya diawali dengan diterimanya sebuah *message* [DHA03].

Untuk obyek-obyek yang memiliki sifat khusus, standar UML mendefinisikan *icon* khusus untuk obyek *boundary, controller* dan *persistant entity* [DHA03].

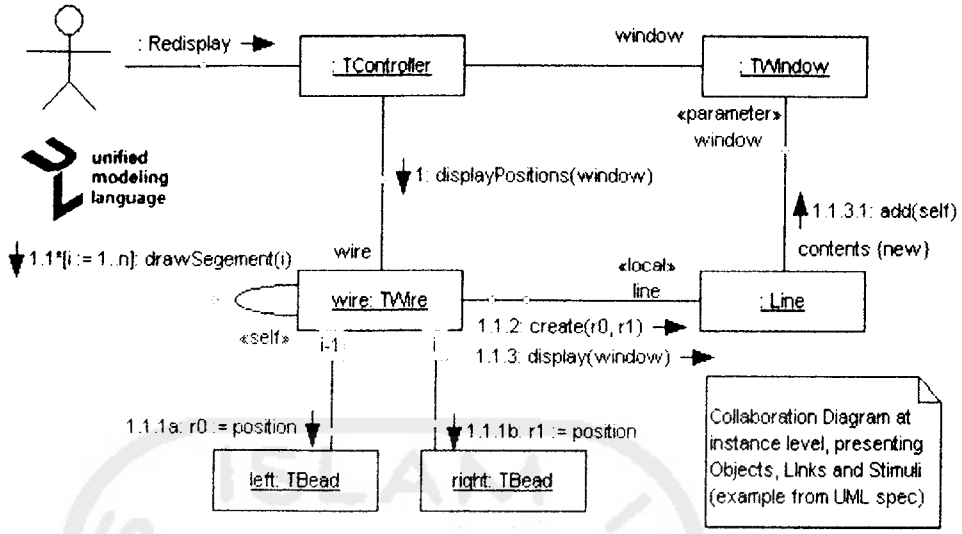
Contoh diagram *sequence* dapat dilihat pada gambar 2.8.



Gambar 2.8 Contoh diagram *sequence* [MOD05]

6. Diagram kolaborasi (*collaboration diagram)*

Diagram kolaborasi juga menggambarkan interaksi antar obyek seperti diagram *sequence*, tetapi lebih menekankan pada peran masing-masing obyek dan bukan pada waktu penyampaian *message*. Setiap *message* memiliki *sequence number*, di mana *message* dari level tertinggi memiliki nomor 1 *Message* dari level yang sama memiliki prefiks yang sama [DHA03]. Contoh diagram kolaborasi dapat dilihat pada Gambar 2.9.

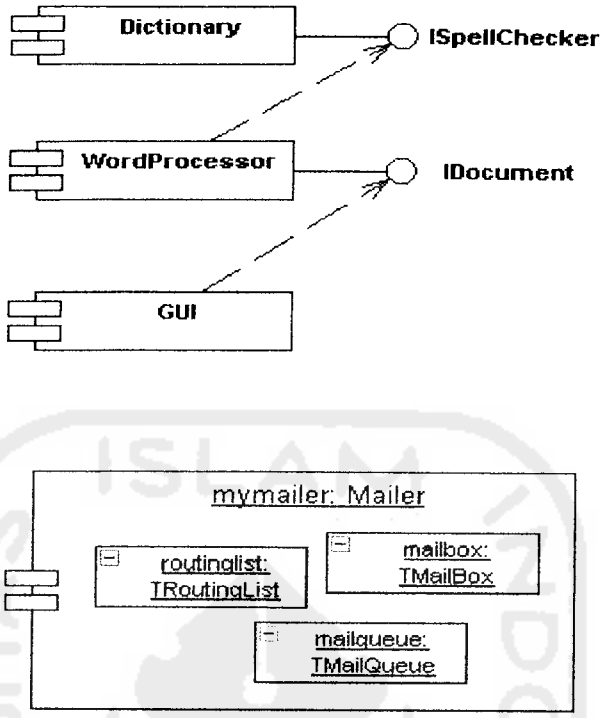


Gambar 2.9 Contoh diagram kolaborasi [MOD05]

7. Diagram komponen (*component diagram*)

Diagram komponen menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) diantaranya [DHA03]. Komponen piranti lunak adalah modul berisi *code*, baik berupa berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time, link time,*maupun *run time.*

Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil. Komponen dapat juga berupa anatrmuka, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain [DHA03]. Contoh diagram komponen dapat dilihat pada Gambar 2.10.



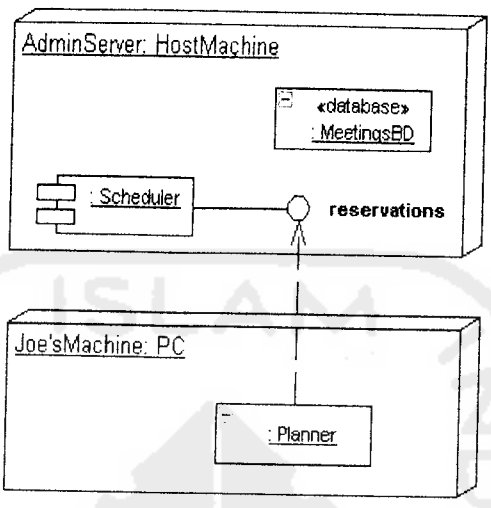
Gambar 2.10 Contoh diagram komponen [MOD05]

8. Diagram *deployment (deployment diagram)*

Diagram *deployment* menggambarkan *detail* bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal lain-lain yang bersifat fisikal [DHA03].

Sebuah *node* adalah server, *workstation*, atau piranti keras yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam

diagram ini [DHA03]. Contoh diagram *deployment* dapat dilihat padfa Gambar 2.11.



Gambar 2.11 Contoh diagram *deployment* [MOD05]

2.5 OpenGL

2.5.1 Definisi

OpenGL adalah suatu spesifikasi standard yang melukiskan suatu *CROSS-LANGUAGE CROSS-PLATFORM API* untuk penulisan aplikasi yang menghasilkan 3D komputer grafik (dan 2D komputer grafik juga). Alat penghubung terdiri dari di atas 250 panggilan fungsi berbeda yang mana dapat digunakan untuk three-dimensional peristiwa ;pemandangan kompleks seri dari yang primitif sederhana. *OpenGL* dikembangkan oleh *Silisium Grafik Inc. (SGI)* pada tahun 1992. *OpenGL*

populer di dalam industri video game di mana/jika bersaing dengan *Direct3D* pada *Microsoft Windows Platform* (lihat *Direct3D* melawan *OpenGL*). *OpenGL* secara luas digunakan di dalam, kenyataan sebetulnya, visualisasi ilmiah, informasi visualisasi, pengembangan game video dan simulasi penerbangan.

2.5.2 Spesifikasi

Pada tingkatan paling dasar nya, *OpenGL* adalah suatu spesifikasi, yang memiliki masud suatu dokumen sederhana yang menguraikan satu set fungsi dan perilaku yang tepat yang harus dikerjakan. Dari spesifikasi ini, *vendors* perangkat keras menciptakan implementasi, suatu fungsi-fungsi yang di ciptakan untuk memenuhi fungsi dinyatakan di dalam spesifikasi *OpenGL*, di mana dimungkinkan untuk menggunakan akselerasi perangkat keras. Perangkat keras harus di test secara spesifik untuk mampu memenuhi persyaratan implementasi mereka sebagai suatu *OpenGL* implementasi.

Implementasi *OpenGL* sangat efisien dinggunakan untuk akselerasi perangkat keras grafis yang *compatible* untuk *Mac O, Windows, Linux, platform Unix*, dan *Playstation 3*. Spesifikasi *OpenGL* ditinjau kembali oleh *OpenGL Architecture Review Board (ARB)* yang dilakukan pada tahun 1992.

2.5.3 Desain

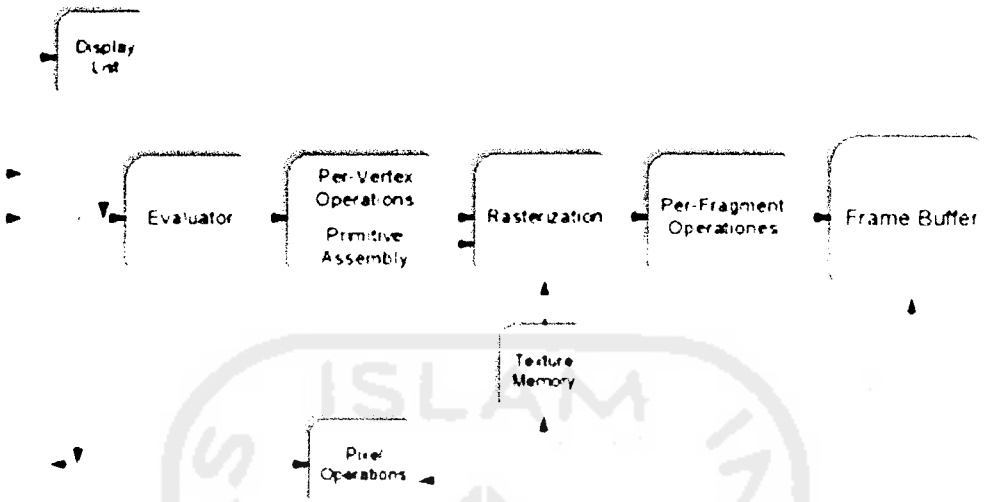
Opengl melayani dua tujuan utama:

* Untuk menyembunyikan kompleksitas dalam interfacing dengan *3D accelerator* berbeda, dengan memperkenalkan programmer dengan tunggal, seragam API.

* Untuk menyembunyikan kemampuan perangkat keras platform yang berbeda, dengan keperluan bahwa semua implementasi mendukung satu set fitur *OpenGL*.

Operasi dasar *OpenGL's* adalah untuk menerima poin-poin, garis dan *polygons* dan mengkonversinya ke dalam pixels. Ini dilakukan oleh suatu saluran grafis/*graphics pipeline* yang dikenal sebagai *OpenGL state machine*. *OpenGL* adalah suatu low-level dalam prosedur API yang menuntut programmer untuk mendikte langkah-langkah yang tepat yang diperlukan untuk memandang suatu peristiwa. Ini kontras dengan deskripsi API, di mana programmer hanya harus menguraikan suatu *scene*/peristiwa dan biarkan fungsi-fungsi di dalamnya mengatur detil dari proses pe-*rendering*-an. Desain *OpenGL* yang *Low-Level* tersebut, memerlukan programmer yang tidak hanya mempunyai pemahaman yang baik tentang saluran grafis/*graphics pipeline*, tetapi juga memberi suatu kebebasan tertentu untuk mengimplementasikan pe-*rendering*-an algoritma. Menurut sejarah *OpenGL* telah mempengaruhi pada pengembangan *3D accelerator*, mempromosikan suatu tingkat dasar dari fungsionalitas yang kini umum di dalam consumer-level perangkat keras :

* Poin-Poin Raster, garis dan *polygons* sebagai dasar,



Gambar 2.12 Proses diagram pipa

* Suatu transformasi dan saluran pencahayaan

* Pemetaan tekstur

* *Alfa Blending*

Suatu uraian ringkas menyangkut proses di dalam saluran grafik dapat diartikan :

1. Evaluasi, jika diperlukan, tentang fungsi polynomial yang menggambarkan masukan tertentu.
2. *verteks operations*, mentransformasikan dan menerangi tergantung pada materialnya.
3. Rasterisasi atau konversi dari informasi yang sebelumnya ke dalam pixels.

4. *Per-Fragment operation*, seperti meng-*update* nilai-nilai yang tergantung pada nilai-nilai kedalaman yang disimpan sebelum atau sesudahnya.
5. Pada akhirnya, fragmen akandisisipkan dalam bingkai *Frame Buffer*.

2.5.4 OpenGL Buffer

Dalam *OpenGL* ada beberapa buffer kunci yang dapat dibahas, yaitu :

- **Buffer akumulasi (*Accumulation Buffer*)**

  Buffer akumulasi berbeda dari buffer lainnya di dalam OpenGL sebab tidak me-*render* *image*/gambar secara langsung. Biasanya, buffer akumulasi digunakan untuk menghasilkan efek khusus seperti *antialiasing* dan gerakan yang *blurring*. Ketika menciptakan cinematics, penyangga/bantalan akumulasi adalah suatu asset penting oleh karena cakupan luasnya, khususnya kemampuan dalam efek khusus.

- **Depth Buffer**

  Pada pemrograman 3D kuno, seorang programmer akan membuat atau menjalankan algoritma dalam waktu riil yang akan mensortir object dari punggung untuk berhadapan, mencegah object dari menarik lebih satu sama lain. Penulisan algoritma ke sort suatu per-frame basis dapat menyingkirkan siklus CPU berharga yang bisa mempersembahkan kepada tugas lain. Yang kebetulan, *depth buffer* menyediakan suatu mekanisme yang *hardware ccelerated* untuk menyumbangkan object dari punggung

untuk berhadapan.. Ada juga suatu fungsi yang dapat menetapkan *depth buffer* mencakup untuk *clipping*. Sekarang ini, hamper semua menggunakan *depth buffer* ketika menulis 3D game. Karena *buffer* ini adalah suatu fungsi penting untuk game.

- **Frame buffer**

  Frame buffer berisi hasil akhir proses pe-*render*an setelah semua fungsi-fungsi buffer yang lain selesai. Hasil dari *frame buffer* biasanya adalah *image* yang ditunjukkan pada layer. *Frame buffer* mempunyai fungsi yang dapat menunjukkan nilai-nilai pixel yang spesifik pada suatu area, tergantung pada kebutuhan. Sayangnya, untuk mengambil nilai-nilai pixel biasanya lambat dan tidak direkomendasikan ketika pe-*rendering-an* dengan kecepatan tinggi diperlukan.

- **Stencil Buffer**

  Stencil buffer menyediakan suatu metoda sederhana untuk me-*render image* ke dalam suatu *cookie-cutter-style*. Kemampuan ini sangat bermanfaat ketika seorang programmer harus memandang seperti pemandangan untuk belakang mobil lewat spion mobil.

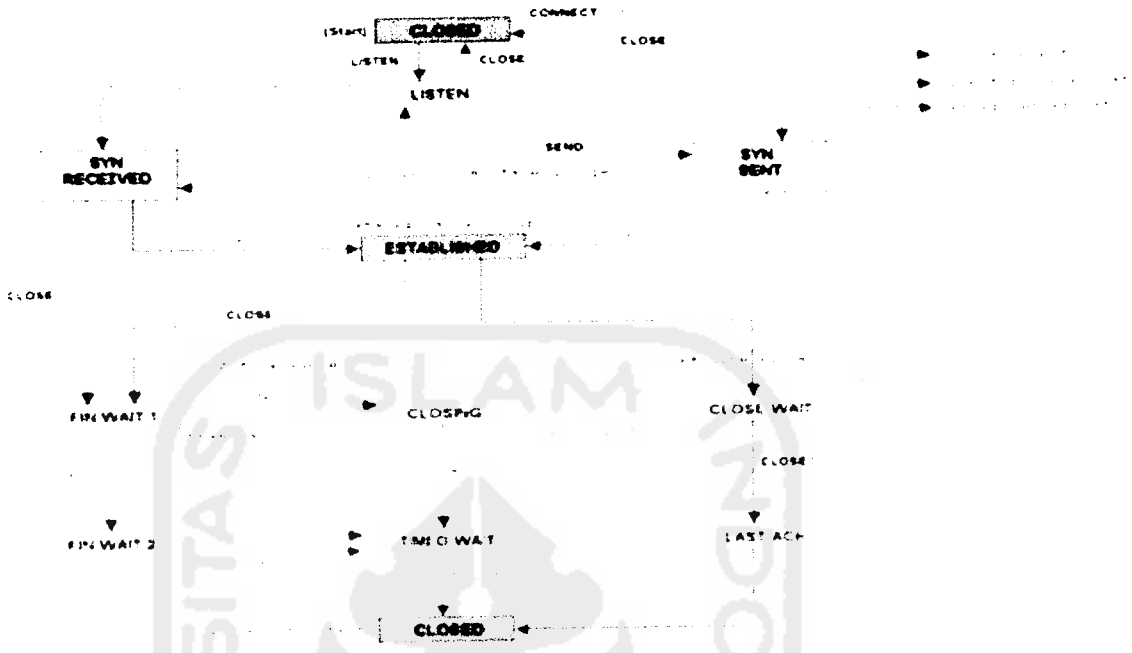
2.6 TCP/IP

2.6.1 Definisi

Transmision Control Protocol (TCP) adalah salah satu dari protokol inti dari deretan Internet protokol, sederhananya dikenal sebagai TCP/IP. Menggunakan TCP, aplikasi pada *network host* dapat menciptakan koneksi ke satu sama lain, di mana mereka dapat menukar arus data yang menggunakan arus stop kontak. Protokol menjamin keaslian dalam penyerahan data dari pengirim ke penerima. TCP juga menciri data untuk hubungan paralel oleh aplikasi berbarengan (contoh, web Server dan e-mail server)yang dijalankan pada host yang sama.

TCP mendukung banyak dari protokol aplikasi internet yang paling populer dan aplikasi hasil, mencakup *World Wide Web* (WWW),e-mail dan *Secure Shell.* Di dalam deretan Internet protokol, TCP menjadi lapisan tengah antara Internet Protokol (IP) di bawah-nya, dan suatu aplikasi di atasnya. Aplikasi lebih sering memerlukan sambungan yang dapat dipercaya ke satu sama lain, sedangkan Internet Protokol tidak menghasilkan arus seperti itu, tetapi lebih hanya upaya terbaik penyerahan. TCP mengerjakan tugas dari lapisan pengangkutan di dalam OSI model jaringan komputer yang disederhanakan. Transport utama yang lain dari internet protokol level adalah UDP.

2.6.2 Protocol Operation



Gambar 2.13 TCP *state diagram*

Tidak sama dengan rekan pendamping tradisional TCP'S, *User Datagram Protocol*, yang mana dapat dengan seketika start mengirimkan paket, TCP menyediakan koneksi yang perlu untuk dibentuk sebelum pengiriman data. koneksi TCP mempunyai tiga fasa:

1.Penetapan koneksi/ *connection establishment*

Untuk menetapkan suatu koneksi, TCP menggunakan suatu *three-way handshake*. Sebelum klien menghubungkan dengan suatu server, server harus terikat lebih dahulu kepada suatu port untuk membuka koneksi.

2. Pemindahan data/ *data transfer*

Ada beberapa kunci menonjolkan TCP [yang] di-set itu terlepas dari *User Datagram Protocol* :

* Ordered data transfer
* Retransmission of lost packets
* Discarding duplicate packets
* Error-free data transfer
* Congestion control

3. Penghentian koneksi/ *connection termination*

BAB III

ANALISIS KEBUTUHAN PERANGKAT LUNAK

3.1 Metode Analisis

Metode Analisis yang dipakai dalam penelitian tugas akhir ini diawali dengan metode pengumpulan data kemudian menganalisa data yang berhasil dikumpulkan dengan metode analisis berarah-obyek.

3.2 Hasil Analisis Kebutuhan

Setelah dianalisis, maka dapat diketahui apa yang menjadi mesukan dan keluaran sistem sesuai dengan apa yang diharapkan dan direncanakan, serta kebutuhan perangkat lunak dan perangkat kerasnya. Hasil analisis adalah berupa bentuk permainan *Othello.*

3.2.1 Kebutuhan Masukan

Adapun data masukan yang dibutuhkan adalah :

1. *Searching* computer yang tersambung.
2. Pemilihan siapa yang akan menjadi *client* dan siapa yang akan menjadi *server.*
3. New Game, dimulainya permainan baru.
4. Meng-klik tempat yang kosong pada papan permainan.

3.2.2 Kebutuhan keluaran

Data keluaran yang akan ditampilkan adalah :

1. Laporan menang, seri, kalah.
2. Score pemain yang akan ditentukan selama permainan.
3. Lama permainan.
4. Adanya konfirmasi apabila jaringan sudah *connect*/tersambung.
5. Adanya keterangan status papan permainan.
6. Peletakan bidak pada posisi awal.

3.2.3 Fungsionalitas yang dikehendaki

a. Fungsi untuk menentukan *client/server.*

b. Fungsi peletakan bidak.

c. Fungsi untukmenunjukkan kondisi atau keadaan pada saat game dijalankan.

d. Fungsi pencahayaan yang muncul dari pointer mouse.

3.3 Analisis Kebutuhan Antar Muka

Kebutuhan terhadap antar muka *(interface)* yang dibuat mempertimbangkan kondisi supaya mudah digunakan oleh pemakai *(user)*. Pembuatan interface ini dibuat atas dasar observasi dan literatur dan software-software yang sudah ada.

Interface yang diinginkan sebaik mungkin sehingga bersifat ramah pengguna *(user friendly)*, artinya pengguna dapat menggunakan perangkat lunak yang dibuat

tidak memberi kesan sulit atau rumit kepada pengguna dengan meminimalkan kesalahan, baik kesalahan masukan, proses maupun keluaran yang dihasilkan disertai dengan umpan balik dari sistem.

3.3.1 Analisis Kebutuhan Perangkat Lunak

Aplikasi pada penelitian tugas akhir ini dikembangkan dengan perangkat lunak Delphi 7 keluaran Borland yang berjalan pada system operasi Microsoft Windows XP Proffesional Service Pack 1.

3.3.2 Analisis Kebutuhan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan aplikasi penelititan tugas akhir ini adalah Notebook Compaq Presario V1027AP dengan spesifikasi sebagai berikut:

a. Processor intel centrino mobile 1.5 GHZ
b. 40GB hard drive
c. 14.1" XGA TFT display
d. Up to 64MB video memory shared
e. DVD CDRW Drive

BAB IV

PERANCANGAN PERANGKAT LUNAK

4.1 Metode Perancangan

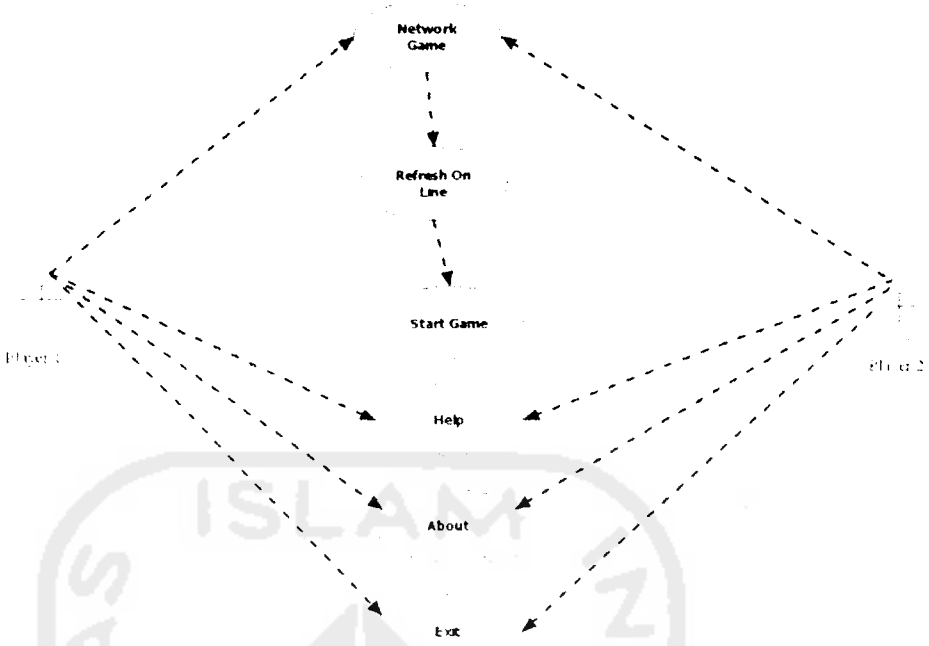
Metode perancangan yang dipakai dalam penelitian tugas akhir ini adalah perancangan berarah-obyek dengan bahasa pemodelan yang diseragamkan (UML). Adapun diagram UML yang digunakan dalam perancangan adalah diagram *use case*, diagram *aktivitas (activity diagram)*, diagram *class (class diagram)* dan diagram *sequence (sequence diagram)*.

4.2 Hasil Perancangan

Hasil perancangan meliputi perancangan antarmuka, serta perancangan model perangkat lunak menggunakan *Unified Modeling Language* (UML).

4.2.1 Diagram *Use Case*

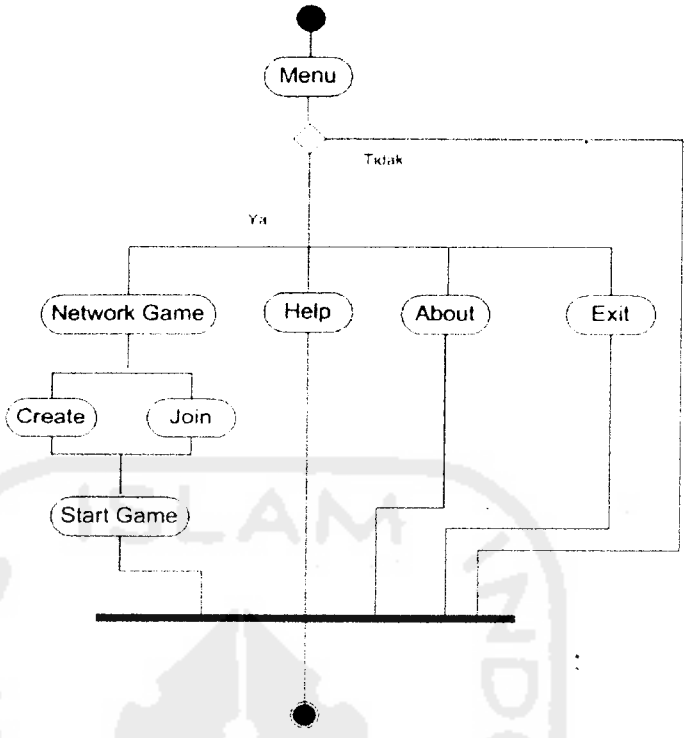
Untuk memulai permainan baru, pada tampilan pertama kali yang diload oleh *user* pada aplikasi *Othello Game* dapat dilakukan seperti yang terlihat pada gambar 4.1.



Gambar 4.1 *Diagram Use Case*

4.2.2 Diagram Aktivitas

Dalam aplikasi *Othello game* hanya ada satu diagram *activity* untuk user.



Gambar 4.2 *Diagram Activity*

4.2.3 Diagram Class

Diagram *class* yang dibangun dalam perangkat lunak akan dikelompokkan ke dalam beberapa kelas, yang fungsinya berbeda satu sama lain sesuai dengan rancangan kelas yang akan dibangun. Dalam aplikasi *Othello game* ini terdapat beberapa kelas yaitu:

a. Class FormMainOtelloNU

Merupakan bagian dari aplikasi yang berisi tampilan awal yang di load pertama kali oleh user. Class ini berisi fungsi-fungsi yang dapat dijalankan pada tampilan awal permainan .

b. Class FormOtelloNconU

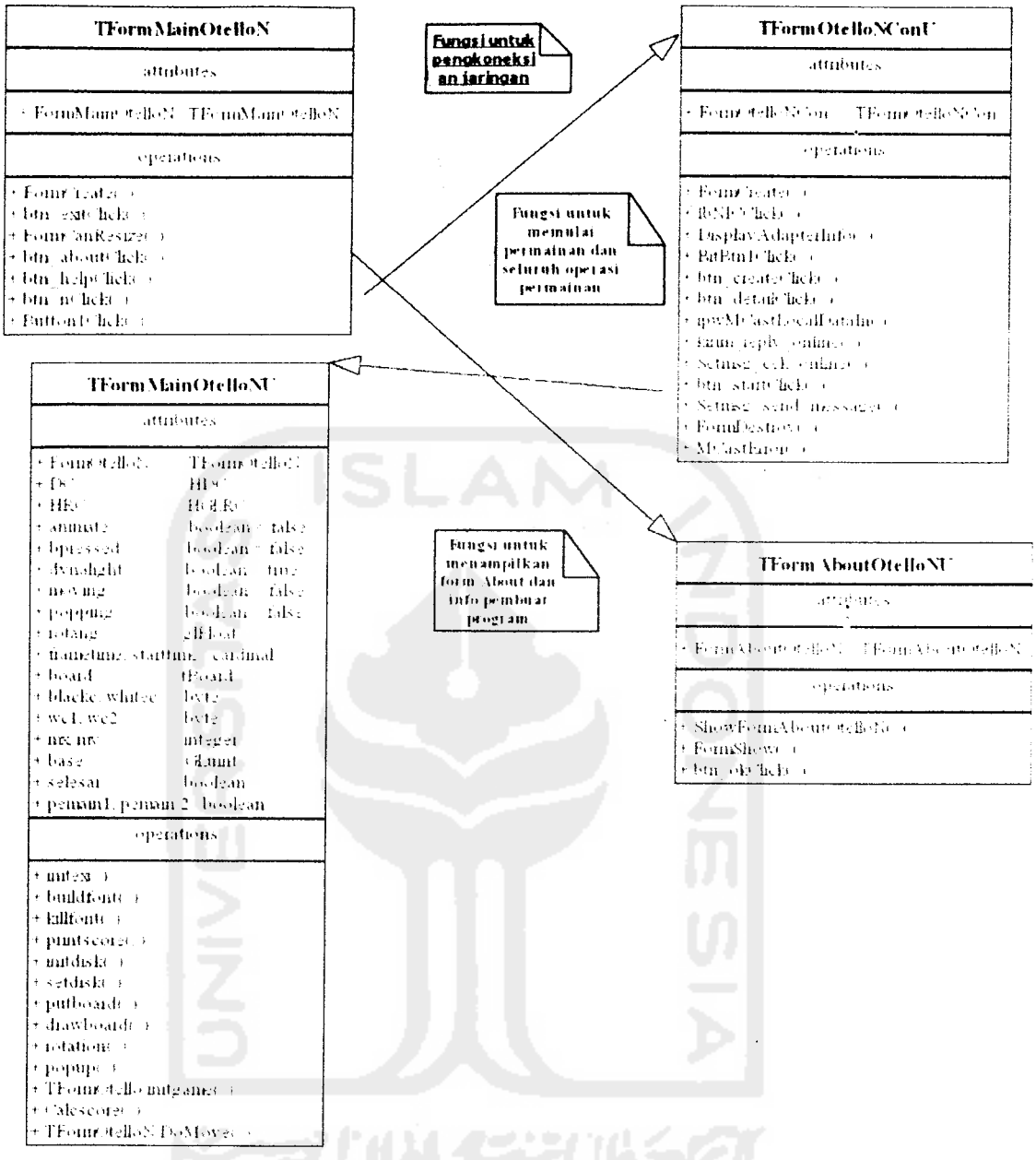
Class ini menjelaskan fungsi-fungsi yang terdapat pada Form Connection Network, yaitu form yang menghubungkan kedua komputer sebelum permainan dapat dimainkan .

c. Class FormOtelloNU

Class yang berisi tentang fungsi-fungsi pada saat permainan dimainkan atau sedang berjalan.

d. Class FormAboutOtelloNU

Class yang berisi fungsi-fungsi untuk menampilkan form about pada permainan Othello.



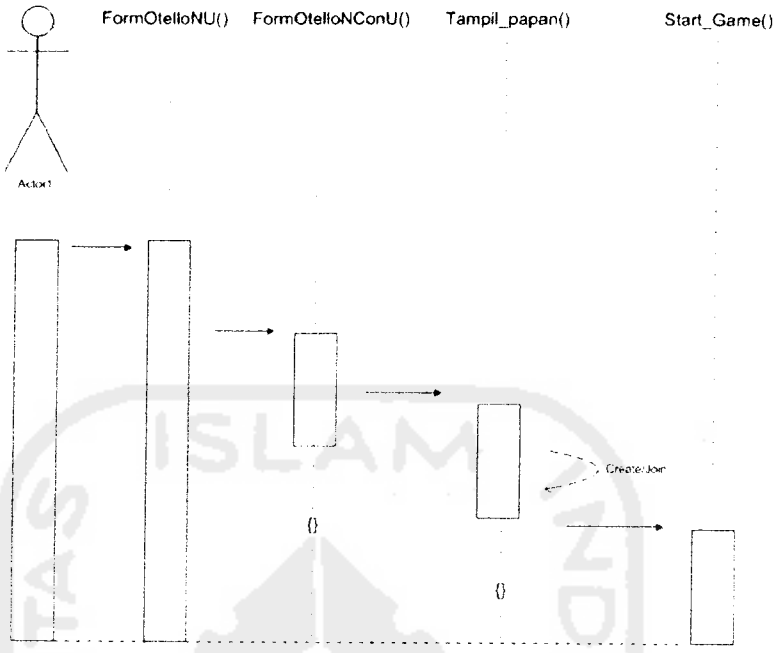
Gambar 4.3 *Diagram Class*

4.2.4 Diagram *Sequence*

Berikut merupakan diagram *sequence* untuk aplikasi *Othello game* :

1. Diagram *sequence Network Game*

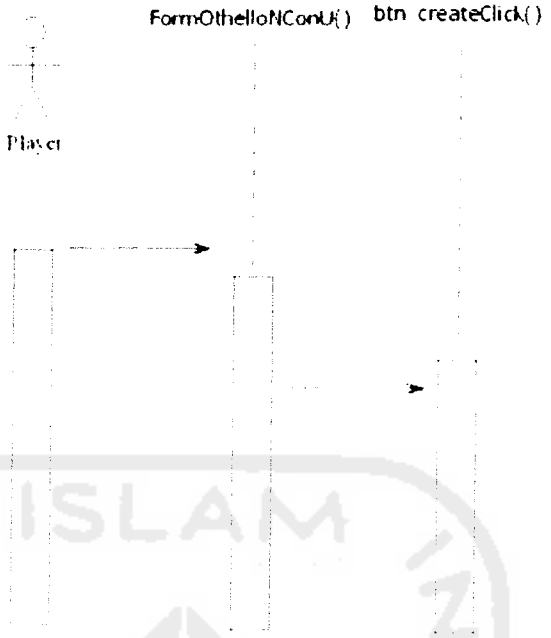
Diagram *sequence Network Game* dapat dilihat pada gambar 4.4



Gambar 4.4 *Diagram Sequence Network Game*

2. Diagram *sequence Refresh Online*

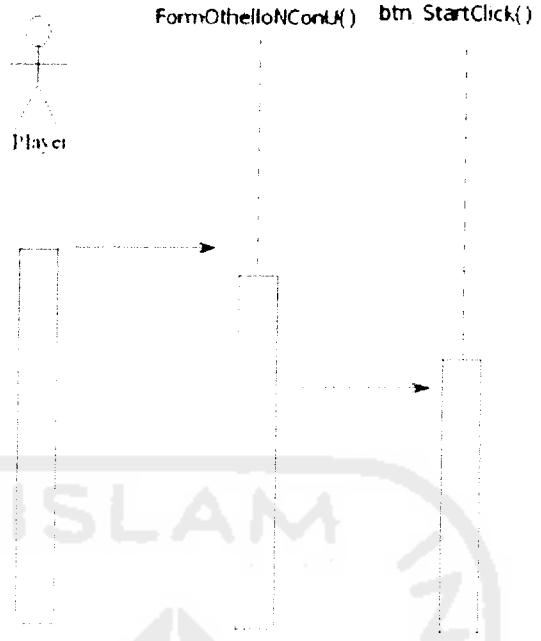
   Diagram *sequence Refresh Online* dapat dilihat pada gambar 4.5



Gambar 4.5 *Diagram Sequence Refresh Online*

3. Diagram *sequence Start Game*

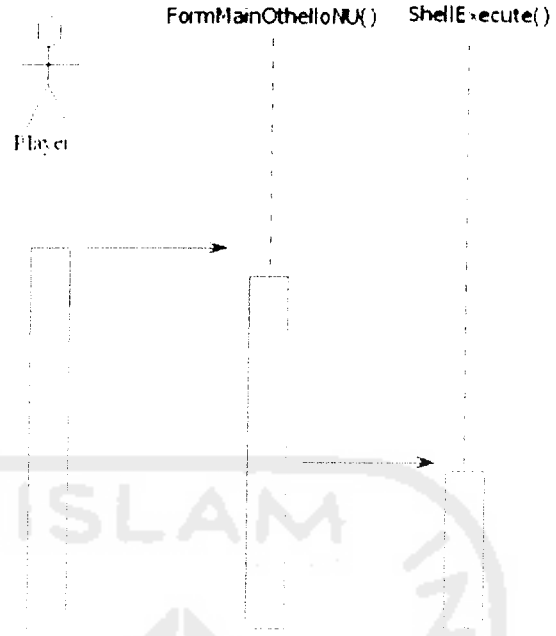
   Diagram *sequence Start Game* dapat dilihat pada gambar 4.6



Gambar 4.6 *Diagram Sequence Start*

4. Diagram *sequence Help*

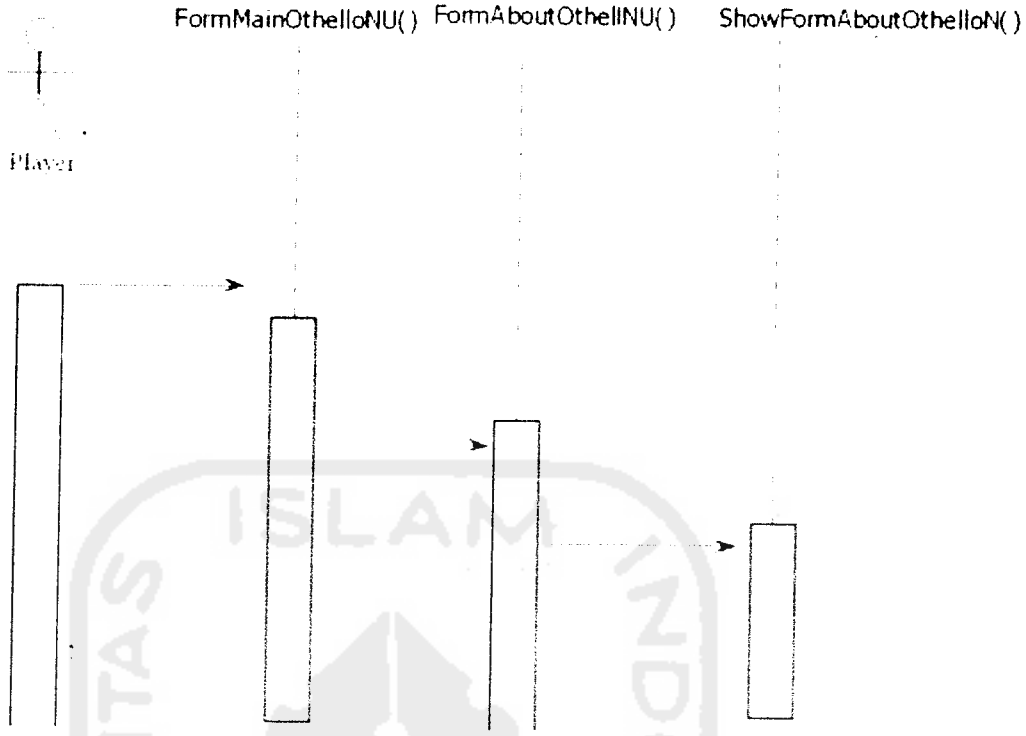
   Diagram *sequence Help* dapat dilihat pada gambar 4.7



Gambar 4.7 *Diagram Sequence Help*

5. Diagram *sequence About*

   Diagram *sequence About* dapat dilihat pada gambar 4.8



Gambar 4.8 *Diagram Sequence About*

6. Diagram *sequence Exit Game*

   Diagram *sequence Exit* dapat dilihat pada gambar 4.9

5. Pe

pa

4 Peranca

Antar m

hello. Rancai

Peranca

Main Fc

Othello

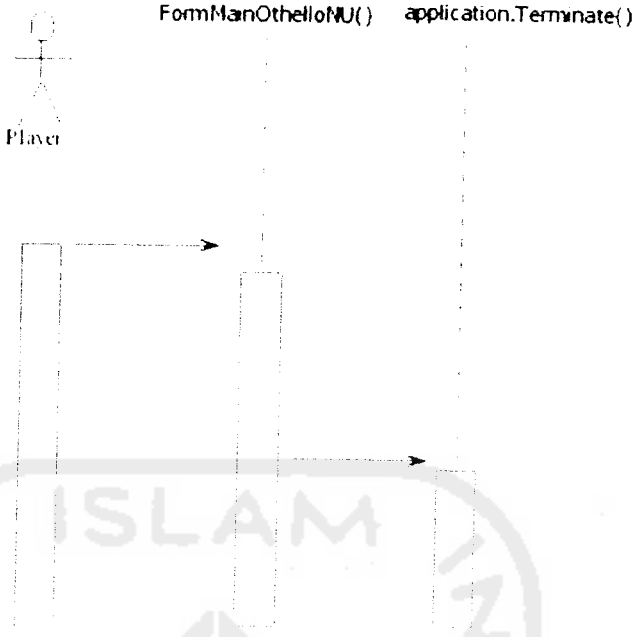
1. Mer

peri

2. Me

pro

3. Me



Gambar 4.9 *Diagram Sequence Exit Game*

4.3 Aturan Bermain

Permainan *Othello* yang akan dirancang dalam penelitian ini mempunyai aturan-aturan sebagai berikut :

1. Pemain melangkah harus melompati buah pemain lawan dan kotak tujuan adalah kotak kosong.
2. Langkah boleh horizontal, vertical dan silang.
3. Nilai masing-masing pemain dihitung berdasarkan buah yang dimiliki masing-masing pemain.
4. Bila salah satu pemain tidak dapat melangkah, maka pemain tersebut haus pass, dan pemain lawannya akan melangkah lagi.

5. Permainan akan berakhir bila kedua pemain tidak dapat melangkah atau papan permainan telah penuh.

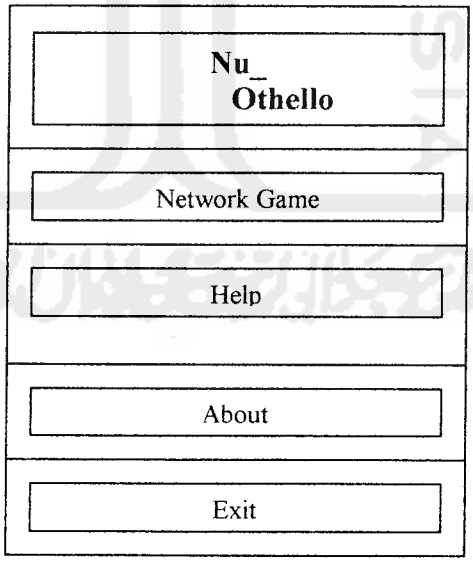
4.4 Perancangan Antar Muka (Interface)

Antar muka dirancang agar pengguna denagn mudah dalam memainkan game Othello. Rancangan antar muka permainan Othello :

a. Perancangan Main Form Game Othello

Main Form adalah form yang di-load pertama kali oleh User. Pada form game Othello local terdapat beberapa bagian, yaitu :

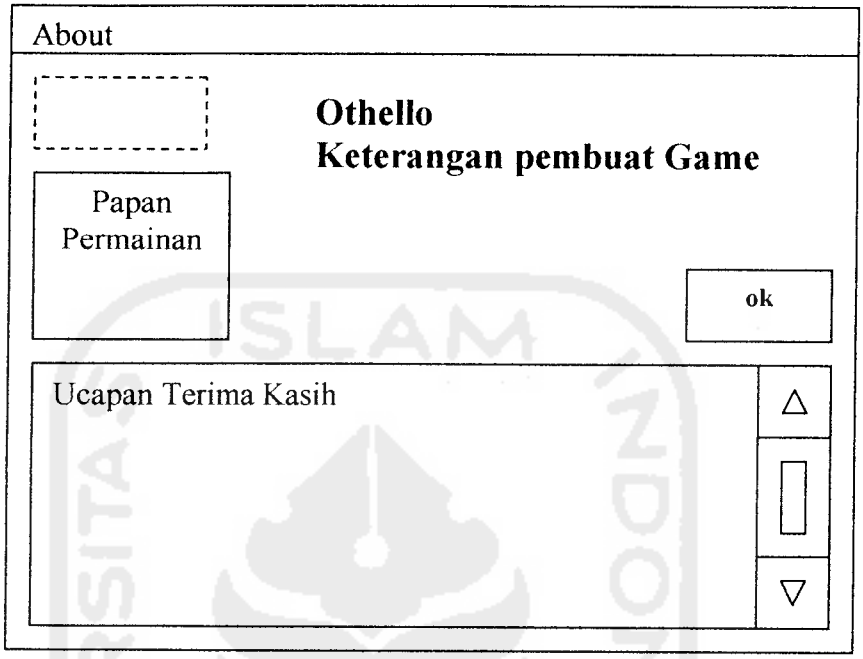
1. Menu atau tombol Network Game, yang berguna untuk memulai permainan.
2. Menu atau tombol Help yang berisi tentang informasi penggunaan program dan aturan-aturan permainan.
3. Menu atau tombol About, yang berisi tentang info pembuat game.



Gambar 410 Rancangan tampilan Main Form

b. Perancangan Form About

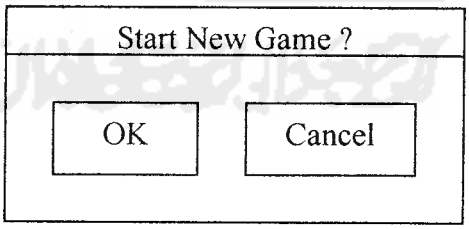
Form about berisi tentang info pembuat game.



Gambar 411 Rancangan tampilan form *about*

c. Rancangan form New Game.

Form Game digunakan untuk memulai permainan baru.

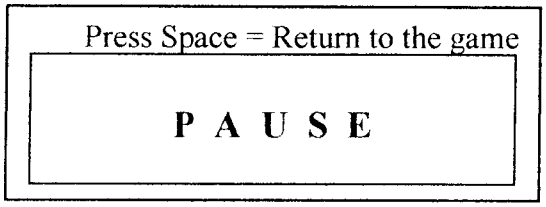


Gambar 4.12 Rancangan tampilan form *new game*

d. Rancangan Tombol Pause

Tombol pause digunakan untuk menghentikan sementara waktu permainan.

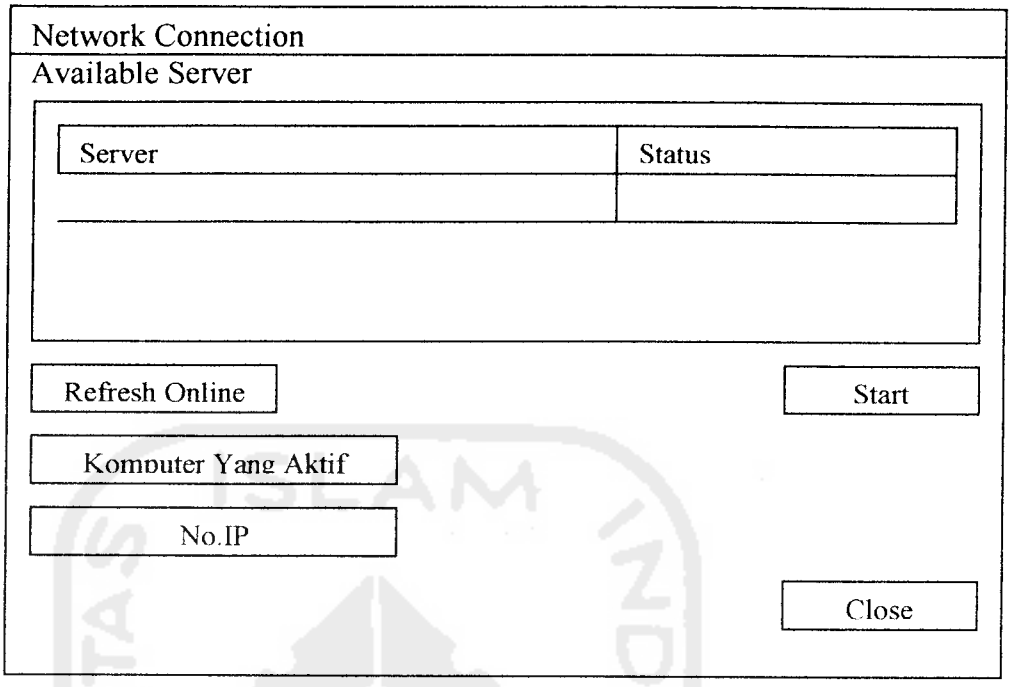
Gambar 4.11



Gambar 4.13 Rancangan tampilan tombol *pause*

e. Rancangan form Network Connection

Form network connection digunakan pada saat akan melakukan permainan multiplayer (maksimal 2 pemain) dengan menggunakan jaringan. Form ini berfungsi untuk mengetahui apakah komputer lain sudah tersambung atau belum, fungsi lainnya adalah untuk menentukan siapa yang jadi *server* dan siapa yang jadi *client*. Gambar 4.12 menunjukkan rancangan form network confirmation

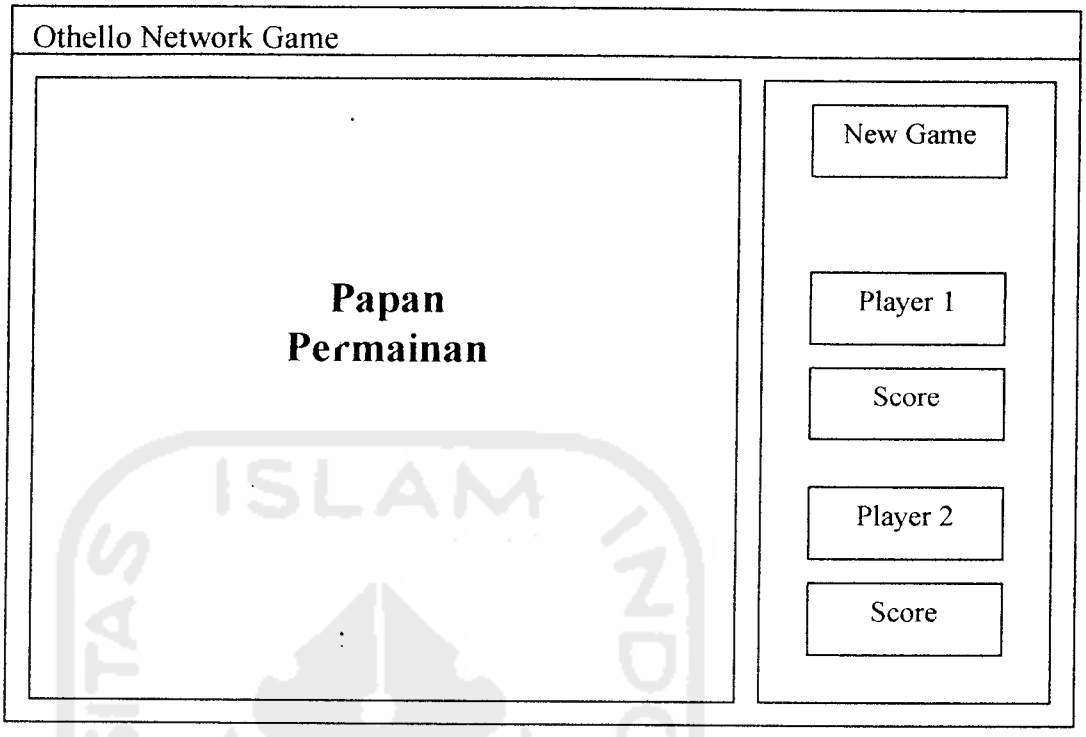


Gambar 4.14 Rancangan Form *Network Connection*

g. Rancangan Form Game Othello Multiplayer

Pada Main form game Othello Multiplayer ada beberapa bagian, yaitu :

1. Score pemain, digunakan untuk mengetahui jumlah buah yang dimiliki oleh tiap pemain.
2. Papan permainan, yang terdiri dari grid-grid kecil yang mempunyai jumlah 8 x 8 kotak
3. Menu New Game , Untuk memulai permainan baru.



Gambar 4.15 Rancangan Form Othello *Network Game*

BAB V

IMPLEMENTASI

5.1 Batasan Implementasi

Pada bagian ini akan menjelaskan apa yang menjadi batasan implementasi perangkat lunak, antara lain : bahasa pemrograman serta alasan pemilihannya, lingkungan pengembangan, asumsi-asumsi yang ditemui dan dibuat selama pengembangan perangkat lunak dan batasan-batasan lain yang juga ditemui selama pengembangan.

Batasan-batasan yang digunakan dalam implementasi aplikasi permainan *Othello* ini adalah sebagai berikut :

5.1.1 Asumsi yang dipakai

Dalam tahap implementasi digunakan asumsi-asumsi sebagai berikut :

1. Aplikasi ini dapat dijalankan pada dua komputer.
2. Aplikasi ini dapat dimainkan oleh dua orang pemain (pemain vs pemain) dalam dua komputer menggunakan jaringan (untuk jenis permainan network game).
3. Pada permainan Othello Network Game yang pertama kali menjalankan permainan adalah pemain yang pertama kali menekan tombol 'refresh online' dan yang menekan tombol 'ready'.

5.1.2 Lingkungan Pengembangan

Lingkungan pengembangan untuk antarmuka dan desain menggunakan lingkungan windows.

5.1.3 Perangkat Lunak yang digunakan

Adapun perangkat lunak yang dipakai untuk membangun permainan *Othello* yaitu:

a. Windows XP Home Edition.

   Sebagai sistem operasi yang dipakai. Sistem ini mendukung bahasa pemrograman yang digunakan dan sangat *user friendly.*

b. Borland Delphi 7.

   Program ini memakai bahasa pemrograman Pascal. Karena telah memliki komponen-komponen visual yang memudahkan dalam interface pada pembuatan *Game Othello.*

5.1.4 Perangkat Keras yang digunakan

Perangkat keras yang digunakan dalam pengembangan aplikasi penelitian tugas akhir ini adalah Notebook Compaq Presario V1027AP denagn spesifikasi sebagai berikut:

a. Processor intel centrino mobile 1,5 GHZ

b. 40GB hard drive

c. 14.1 " XGA TFT display

d. Up to 64MB video memory shared

e. DVD CDRW Drive.

5.2 Implementasi

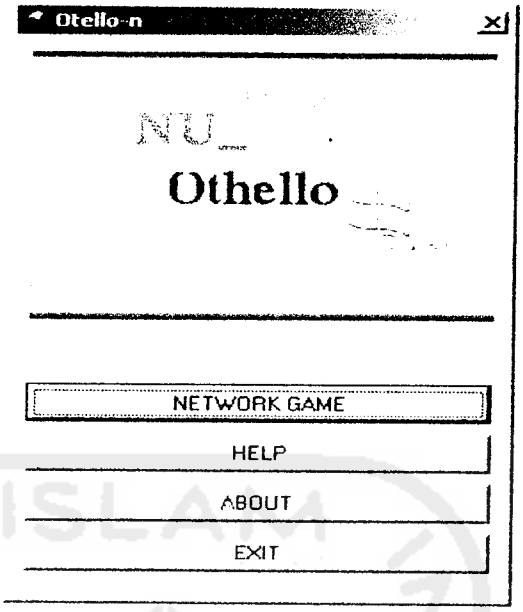
Secara garis besar implementasi aplikasi permainan *Othello* adalah menyusun strategi untuk menempatkan buah-buah dalam papan permainan. Implementasi aplikasi permainan *Othello* terbagi menjadi 2 bagian yaitu : Implementasi Interface dan Implementasi Source Code.

5.2.1 Implementasi Interface

Implementasi interface ini berisi tentang gambar dari menu–menu yang ada dalam permainan Othello, menu-menu tersebut diantaranyan adalah sebagai berikut:

1. **Menu Utama**

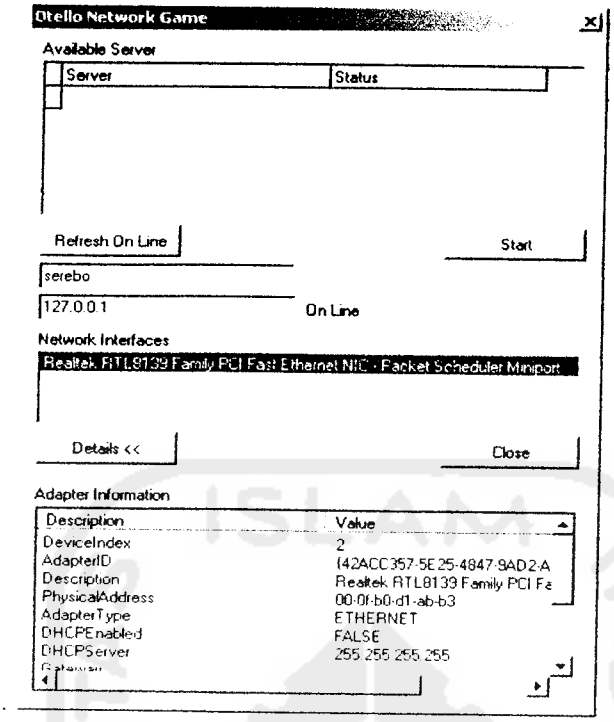
   Ketika aplikasi permainan *Othello* pertama kali dijalankan maka akan tampil gambar yang disebut form utama seperti pada gambar 5.1, Tampilan ini berisi menu utama yang menghubungkan dengan form lainnya, menu utama terdiri dari Network Game, help, about. dan exit.



Gambar 5.1 Menu Utama

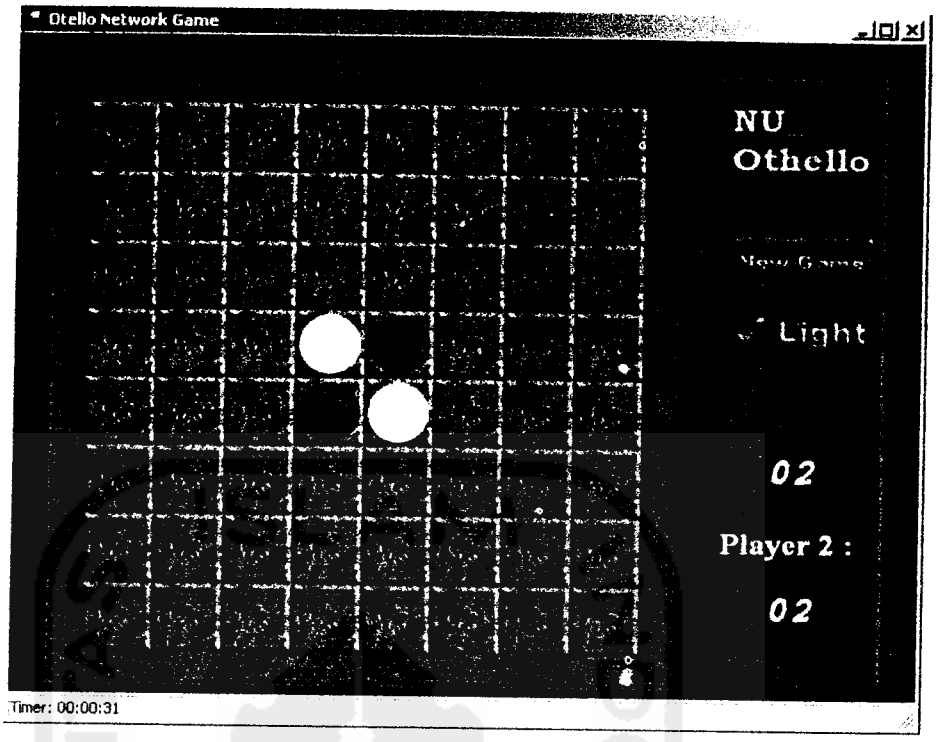
2. Menu Network Game

Untuk menampilkan Network Game dibutuhkan suatu form yang menunjukkan koneksi dalam game tersebut, form tersebut bernama Form Connection Network.



Gambar 5.2 Menu Connection Network

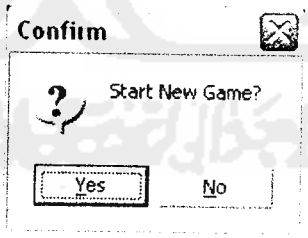
Setelah muncul form tersebut, langkah selanjutnya adalah dengan menekan tombol "Refresh Online", dimana tombol tersebut berguna untuk men-scan apakah ada komputer lain yang online. Setelah tersambung maka permainan Othello siap untuk dimainkan. Papan permainan akan muncul seperti pada gambar 5.3.



Gambar 5.3 Papan Permainan Othello Network Game

3. Menu New Game

Menu New Game digunakan untuk memulai baru permainan, seperti terlihat pada gambar 5.4.



Gambar 5.4 Menu New Game

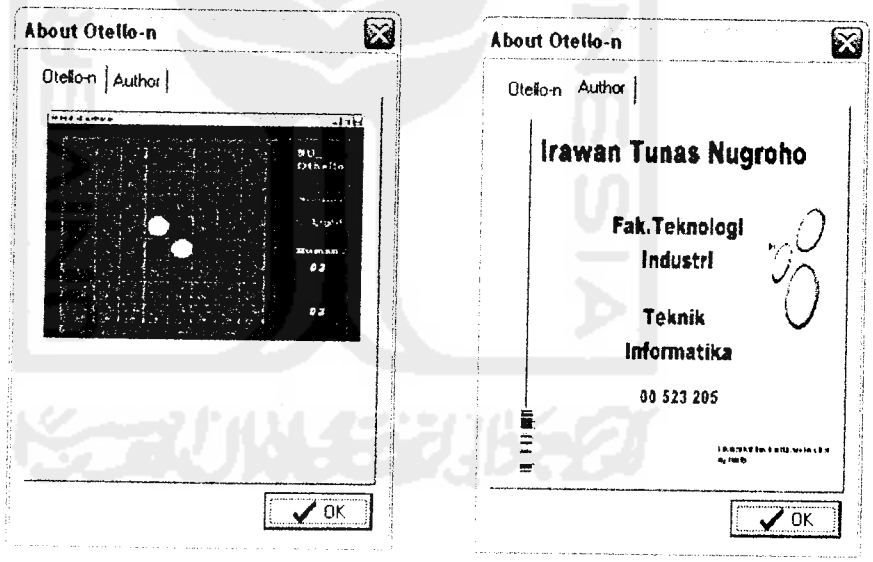
4. Menu Pause

User dapat menghentikan waktu permainan dengan menekan tombol Pause, dan untuk kembali memainkan game user dapat menggunakan tombol spasi pada keyboard. Gambar 5.5 adalah tampilan disaat dame dalam keadaan pause.

Gambar 5.5 Menu Pause

5. Menu About

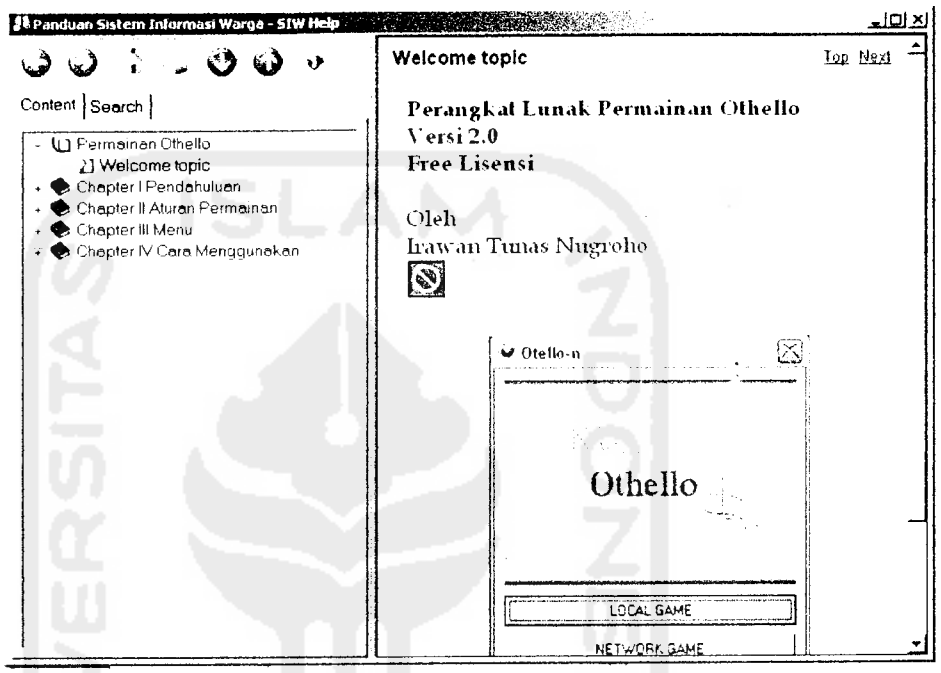
Tampilan menu About ini berisi tentang profil pembuat program, seperti terlihat pada gambar 5.6



Gambar 5.6 Menu About

6. Menu Help

Pada tampilan Help ini digunakan untuk membantu *user* menggunakan program dan aturan-aturan dalam permainan *Othello*, seperti terlihat pada Gambar 5.7.

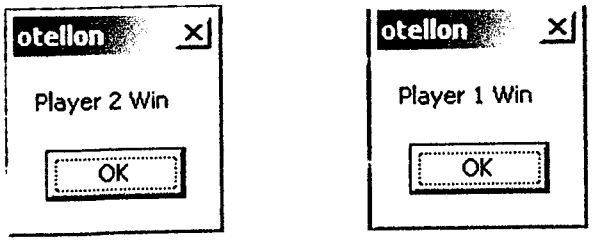


Gambar 5.7 Menu Help

7. Tampilan Akhir Permainan

Pada Akhir permainan akan ditampilkan pesan, pesan yang ditampilkan sesuai dengan banyaknya buah yang diperoleh selama pemainan oleh pemain, (human vs computer) atau pemain1 vs pemain2 untuk model permainan network, tampilan pesan seperti terlihat dibawah ini:

a. Tampilan akhir permainan Othello Network



Gambar 5.8 Tampilan akhir Othello Network Game

5.2.2 Implementasi Procedural

Implementasi procedural berisi tentang prosedur–prosedur dan fungsi–fungsi yang dijalankan pada permainan Othello, implementasi tersebut adalah sebagai berikut:

1. **Prosedur untuk menampilkan form Connection Network**

   Prosedur ini digunakan untuk menampilkan form connection network, yaitu form yang berfungsi untuk men-scan jaringan apakah ada computer yang tersambung.

```
procedure TFormOtelloNCon.FormCreate(Sender: TObject);
var
  i                 : integer;
begin
  //sembunyikan detail
  btn_detailClick(btn_detail);
  btn_create.Enabled := false;
  //letak form ini
  self.Top := FormMainOtelloN.Top + (FormMainOtelloN.Height
div 2);
  self.Left := FormMainOtelloN.Left;
 //nama server adalah nama computer
  txt_server.Text := MCast.LocalHost;

  //informasi komputer jaringan
  for i := 1 to ipinfo.AdapterCount do
    begin
      ipinfo.AdapterIndex := i;
```

```
      lbNIC.Items.Add(Ipinfo.AdapterDescription);
    end;
  if lbNIC.Items.Count > 0 then
    begin
      lbNIC.ItemIndex := 0;
      DisplayAdapterInfo();
    end;

  //konstanta komuniksi
  otello_port := 9191;
  svr_otello_port := 9191;
  lbNICClick(lbNIC);
end;
```

2. Prosedur untuk New Game atau memulai permainan baru

Prosedur ini digunakan untuk memulai permainan baru.

```
if MessageDlg('Start New Game?', mtConfirmation, [mbyes,
mbno], 0) = MrYes then
      if FormPemain.new_game then
        begin
          selesai := false;
          FormOtelloLocal.Visible := true;
          initgame;
          FormOtelloLocal.fstart_time := now();
          FormOtelloLocal.TimerGame.Enabled := true;
        end
      else
        begin
          //FormOtelloLocal.Visible := true;
          FormOtelloLocal.Close;
        end;
    end;
end;
```

3. Prosedur untuk tombol Pause

Prosedur yang dijalankan untuk menunda sementara permainan dengan cara menekan tombol 'spasi'.

```
procedure ShowPause(theTimer: TTimer; formInduk: TForm);
begin
  try
    FormPause := TFormPause.Create(Application);
    // posisi form pause
```

```
    FormPause.Top  :=  formInduk.Top  +  ((formInduk.Height  -
FormPause.Height) div 2);
    FormPause.Left  :=  formInduk.Left  +  ((formInduk.Width  -
FormPause.Width) div 2);
theTimer.Enabled := false;
   FormPause.ShowModal;
    theTimer.Enabled := true;
  finally
    FormPause.Free;
  end;
end;
```

4. Prosedur untuk tampilan akhir permainan

Prosedur ini dijalankan atau digunakan pada saat menampilkan keterangan atau tampilan akhir permainan berupa pesan menang atau kalah.

```
 if selesai then
     begin


      //FormOtelloN.Visible := false;
      FormOtelloN.TimerGame.Enabled := false;

      if blackc > whitec then
       begin
         //FormPemain.lbl_waktu_game.Caption     :=     'Time
Elapsed : ' + FormatDateTime('HH:NN:SS', FormOtelloN.fgame_time);
         //FormPemain.ShowModal;
         ShowMessage('Player 1 Win');
         exit;
       end
      else
      if blackc < whitec then
       begin
         showMessage('Player 2 Win');
         exit;
       end
      else
      if blackc = whitec then
       begin
        showmessage('Draw !');
        exit;
       end;
    end;
```

5. Prosedur untuk menghitung score

Prosedur ini digunakan untuk mengitung jumlah bidak atau menampilkan score pada papan permainan.

```
procedure CalcScore;
var
  i, j              : integer;
begin
  blackc := 0;
  whitec := 0;
  for i := 0 to 7 do
    for j := 0 to 7 do
      begin
        if (board[j, i] = 1) then
          inc(whitec);
        if (board[j, i] = 2) then
          inc(blackc);
      end;
end;
```

6. Fungsi Untuk membalik buah

Fungsi ini untuk kedua pemain yaitu. Fungsi yang digunakan disini merupakan fungsi pengecekan delapan tetangga. Fungsi ini akan terus diproses sampai akhir, sampai ditemukan posisi buah yang akan dibalik.

a. Perintah untuk pengecekan kiri

Perintah ini dijalankan untuk men-check 8 tetangga dari bidak yang sudah ada, dalam hal ini fungsi dijalankan apakah pada sebelah kiri bidak memungkinkan untuk diisi bidak selanjutnya.

```
test := FALSE;
for i := cx - 1 downto 0 do
  begin
    if (board[cy, i] = wc1) then
      test := TRUE
    else
      if ((board[cy, i] = wc2) and (test)) then
      begin
        passed := TRUE;
```

```
          if not check then
            for j := cx - 1 downto i + 1 do
              board[cy, j] := 3;
          break;
        end
      else
        break;
    end;
```

b. Perintah untuk pengecekan kanan

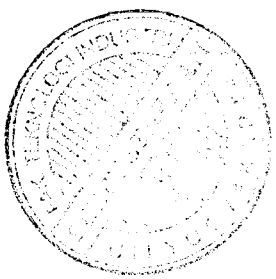
Perintah ini dijalankan untuk men-check 8 tetangga dari bidak yang sudah ada, dalam hal ini fungsi dijalankan apakah pada sebelah kanan bidak memungkinkan untuk diisi bidak selanjutnya.

```
test := FALSE;
for i := cx + 1 to 7 do
begin
  if (board[cy, i] = wc1) then
    test := TRUE
  else
    if ((board[cy, i] = wc2) and (test)) then
    begin
      passed := TRUE;
      if not check then
        for j := cx + 1 to i - 1 do
          board[cy, j] := 3;
        break;
    end
  else
    break;
end;
```

c. Perintah untuk pengecekan atas

Perintah ini dijalankan untuk men-check 8 tetangga dari bidak yang sudah ada, dalam hal ini fungsi dijalankan apakah pada sebelah atas bidak memungkinkan untuk diisi bidak selanjutnya.

```
test := FALSE;
for i := cy - 1 downto 0 do
  begin
    if (board[i, cx] = wc1) then
     test := TRUE
```



```
    else
      if ((board[i, cx] = wc2) and (test)) then
      begin
        passed := TRUE;
        if not check then
          for j := cy - 1 downto i + 1 do
            board[j, cx] := 3;
        break;
      end
    else
      break;
  end;
```

d. Perintah untuk pengecekan bawah

Perintah ini dijalankan untuk men-check 8 tetangga dari bidak yang sudah ada, dalam hal ini fungsi dijalankan apakah pada sebelah bawah bidak memungkinkan untuk diisi bidak selanjutnya.

```
test := FALSE;
for i := cy + 1 to 7 do
  begin
    if (board[i, cx] = wc1) then
      test := TRUE
    else
      if ((board[i, cx] = wc2) and (test)) then
      begin
        passed := TRUE;
        if not check then
          for j := cy + 1 to i - 1 do
            board[j, cx] := 3;
        break;
      end
    else
      break;
  end;
```

e. Perintah untuk pengecekan kiri atas

Perintah ini dijalankan untuk men-check 8 tetangga dari bidak yang sudah ada, dalam hal ini fungsi dijalankan apakah pada sebelah kiri atas bidak memungkinkan untuk diisi bidak selanjutnya.

```
test := FALSE;
```

```
for i := 1 to 7 do
  begin
    if (((cy - i) >= 0) and ((cx - i) >= 0)) then
      if (board[cy - i, cx - i] = wc1) then
       .test := TRUE
      else
        if ((board[cy - i, cx - i] = wc2) and (test)) then
         begin
           passed := TRUE;
           if not check then
             for j := 1 to i - 1 do
                board[cy - j, cx - j] := 3;
           break;
         end
      else
        break
   else
     break;
  end;
```

f. Perintah untuk pengecekan kiri bawah

Perintah ini dijalankan untuk men-check 8 tetangga dari bidak yang sudah ada, dalam hal ini fungsi dijalankan apakah pada sebelah kiri bawah bidak memungkinkan untuk diisi bidak selanjutnya.

```
test := FALSE;
for i := 1 to 7 do
  begin
    if (((cy + i) < 8) and ((cx - i) >= 0)) then
      if (board[cy + i, cx - i] = wc1) then
        test := TRUE
      else
        if ((board[cy + i, cx - i] = wc2) and (test)) then
        begin
          passed := TRUE;
          if not check then
            for j := 1 to i - 1 do
               board[cy + j, cx - j] := 3;
          break;
        end
      else
        break
    else
      break;
```

```
end;
```

g. Perintah untuk pengecekan kanan atas

Perintah ini dijalankan untuk men-check 8 tetangga dari bidak yang sudah ada, dalam hal ini fungsi dijalankan apakah pada sebelah kanan atas bidak memungkinkan untuk diisi bidak selanjutnya.

```
test := FALSE;
for i := 1 to 7 do
  begin
    if (((cy - i) >= 0) and ((cx + i) < 8)) then
      if (board[cy - i, cx + i] = wc1) then
        test := TRUE
      else
        if ((board[cy - i, cx + i] = wc2) and (test)) then
        begin
          passed := TRUE;
          if not check then
            for j := 1 to i - 1 do
              board[cy - j, cx + j] := 3;
          break;
        end
      else
        break
    else
      break;
  end;
```

h. Perintah untuk pengecekan kanan bawah

Perintah ini dijalankan untuk men-check 8 tetangga dari bidak yang sudah ada, dalam hal ini fungsi dijalankan apakah pada sebelah kanan bawah bidak memungkinkan untuk diisi bidak selanjutnya.

```
test := FALSE;
for i := 1 to 7 do
  begin
    if (((cy + i) < 8) and ((cx + i) < 8)) then
      if (board[cy + i, cx + i] = wc1) then
        test := TRUE
      else
      if ((board[cy + i, cx + i] = wc2) and (test)) then
          begin
```

```
            passed := TRUE;
            if not check then
              for j := 1 to i - 1 do
                board[cy + j, cx + j] := 3;
            break;
          end
        else
          break
      else
        break;
    end;
```

BAB VI

ANALISIS KINERJA PERANGKAT LUNAK

6.1 Analisis Kesesuaian

Sesuai dengan teori dalam pembuatan program permainan pada bab 2 analisis yang dilakukan terhadap *GAME WORLDS* atau elemen pada program *game Othello* yaitu:

1. Game Board

   Program menggunakan tampilan 3D dengan kesesuaian antara latar belakang, amage biji dan papan.

2. Instruksi untuk pemain

   Program telah dilengkapi dengan keterangan cara bermain dan aturan bermain yang dapat dipanggil sewaktu-waktu.

3. Informasi untuk pemain

   Informasi pada game Othello ini berupa waktu permainan, keping pemain dan score pemain.

4. Penghargaan

   Penghargaan akan diberikan kepada pemain yang telah memenangkan permainan Othello. Penghargaan ini berupa ucapan selamat kepada pemain tersebut.

5. Variasi

   Game ini dapat dimainkan oleh dua pemain pada dua komputer dengan menggunakan jaringan, hal ini membuat permainan tidak monoton dan akan berjalan lebih menarik.

6.1.1 Analisis Kesesuaian perhitungan data pada koordinat papan

Tiap koordinat papa memiliki prioritas nilai. Prioritas nilai yang akan menjadi koordinat jalan terbaik menurut fungsi pengecekan 8 tetangga dan perhitungan informasi data. Untuk mengetahui program sudah menjalankan pengecekan 8 tetangga, perhitungan data dan pemilihan jalan yang sesuai dengan skema prioritas koordinat maka akan dilakukan beberapa analisis kesesuaian perhitungan.

Peta koordinat prioritas ditunjukkan oleh gambar 6.1 dibawah ini.

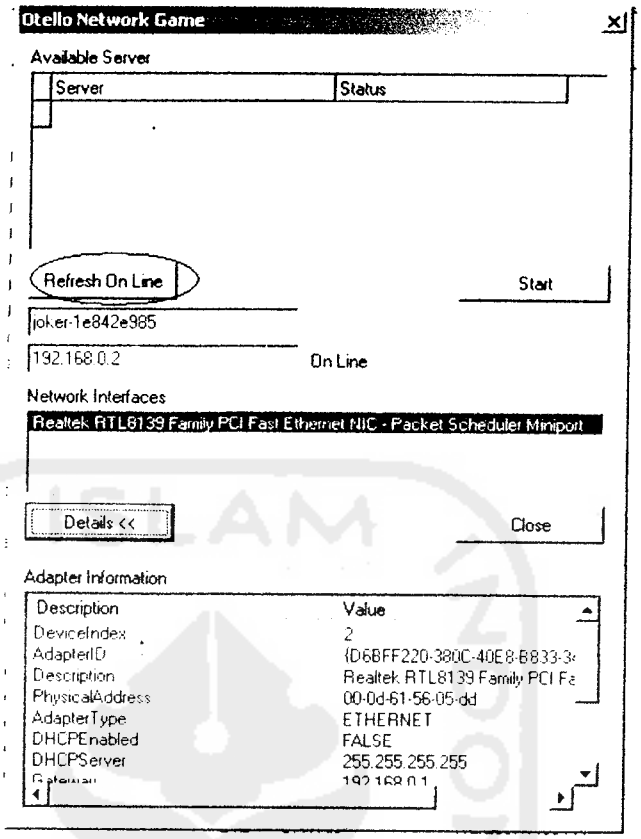
1. Untuk yang berwarna merah merupakan koordinat prioritas pertama yang dinamakan sudut
2. Untuk yang berwarna kuning merupakan koordinat prioritas kedua yang dinamakan tepi.
3. Untuk yang berwarna biru merupakan koordinat prioritas ketiga yang dinamakan tengah
4. Untuk yang berwarna hijau merupakan koordinat prioritas yang dinamakan kotak 2x2

(0,0)	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)	(0,6)	(0,7)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)	(2,7)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)	(3,7)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)	(4,7)
(5,0)	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)	(5,7)
(6,0)	(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)	(6,7)
(7,0)	(7,1)	(7,2)	(7,3)	(7,4)	(7,5)	(7,6)	(7,7)

Gambar 6.1 Skema koordinat prioritas

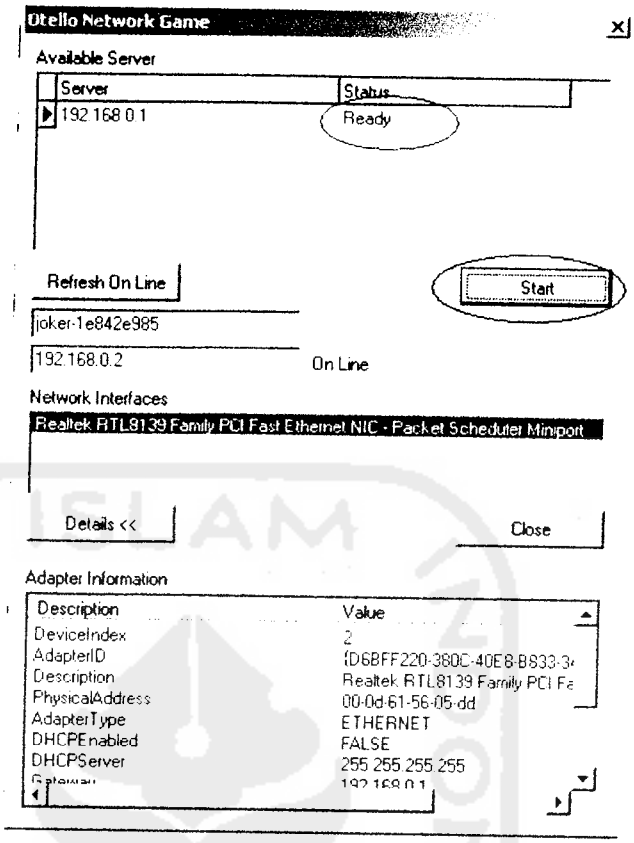
6.2 Pengujian Program

Setelah pada menu utama dipilih model permainan network game, maka yang muncul setelah itu adalah form connection. Langkah awal untuk memulai permainan adalah dengan cara menekan tombol 'refresh online', diman tombol ini berfungsi untuk men-scan apakah jaringan atau port komputer lain sudah terdeteksi atau sudah tersambung, gambar 6.2 akan menunjukkan tombol refresh online dalam form connection network.



Gambar 6.2 Letak tombol refresh online

Setelah tombol refresh online ditekan maka computer akan mulai untuk men-scan port lain yang terhubung, setelah sudah terhubung maka akan muncul keterangan bahwa computer lain sudah terhubung dan siap untuk memulai permainan. Hal ini ditandai dengan adanya keterangan 'ready' pada form. Maka langkah selanjutnya adalah dengan menekan tombol 'Start' yang berada sejajar dengan tombol refresh online. Gambar 6.3 menunjukkan keterangan bahwa koneksi sudah tersambung dan menunjukkan letak tombol Start.



Gambar 6.3 Jaringan tersambung & Letak tombol Start

Baru setelah semua langkah diatas dijalankan, maka akan muncul papan permainan yang siap untuk dimainkan oleh dua orang. Dalam Network Game ini pemain pertama yang menjalankan permainan adalah pemain yang pertama menekan tombol 'Refresh Online' dan tombol 'Start'. Permainan ini dijalankan secara bergantian oleh kedua pemain sampai permainan selesai, yaitu pada saat papan permainan telah penuh atau salah satu pemain sudah tidak mampu untuk melangkah lagi.

6.2.1 Pengujian Pada Pemain

Untuk mengetahui apakah program yang telah dibuat ini dapat berjalan baik, maka program akan diujikan kepada beberapa pemain. Dalam hal ini pengujian akan dilakukan pada 2 pemain yang memainkan permainan Othello sengan cara saling melawan. Pada pengujian ini menggunakan asumsi bahwa pengujian akan dilakukan sebanyak 5 kali.

Pengujian pada beberapa pemain dapat dilihat pada tabel 6.1.

Tabel 6.1 Pengujian pada 2 pemain

No	Nama	Ronde 1	Ronde 2	Ronde 3	Ronde 4	Ronde 5	Jumlah Menang
		Jumlah bidak	Jumlah bidak	Jumlah bidak	Jumlah bidak	Jumlah bidak	
1	Antok	42	44	31	29	49	3
2	Efri	22	20	33	35	15	2

Dari Tabel 6.1 dapat dilihat pengujian program yang dilakukan oleh 2 orang pemain. Pada pengujian yang dilakukan sebanyak 5 kali, dapat dilihat bahwa pemain 1 (Antok) berhasil memenangkan sebanyak 3 kali sedangkan pemain 2 (Efri) menang sebanyak 2 kali.

6.3 Hasil Analisis

Dari Pengujian yang telah dilakukan maka dapat didapat hasil pengujian sebagai berikut :

- Hasil Analisis untuk Othello Network Game

1. Sistem telah mampu melakukan koneksi antar jaringan, sehingga permainan dapat dilakukan dengan menggunakan dua komputer.
2. Sistem talah mampu menampilkan nilai, pesan kesalahan dan pesan menang atau kalah bagi tiap-tiap pemain.

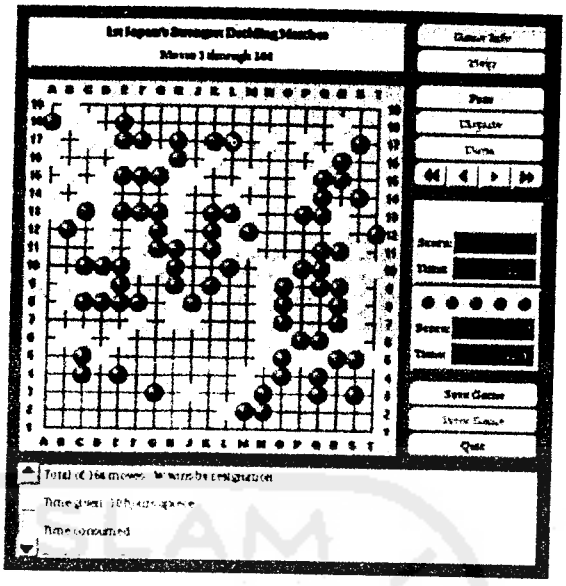
6.3.1 Perbandingan dengan game Othello yang sudah pernah ada

Hasil dari perancangan game Othello ini kemudian dibuat perbandingan dengan game Othello yang sudah pernah dibuat, dimana perbandingan tersebut berguna untuk pengembangan game Othello lebih lanjut. Tabel perbandingan dapat dilihat pada tabel 6.2

Tabel 6.2 Perbandingan dengan game Othello yang sudah pernah dirancang

Othello yang sudah ada		**Othello yang dibuat**	
1	Dijalankan pada sistem operasi Linux	1	Dijalankan pada sistem operasi Windows
2	Permainan secara jaringan, Game harus connect ke internet bukan intranet	2	Permainan secara jaringan hanya menggunakan intranet.
3	Dapat mengubah-ubah tampilan papan permainan	3	Tidak dapat mengubah tampilan papan permainan
4	Dapat bertindak sebagai jembatan Go modem protocol	4	Tidak dapat bertindak sebagai Go modem protocol

Tabel diatas diperoleh dari perbandingan game Othello yang dibuat dengan game Othello yang bernama CGoban. Dimana game ini dapat dilihat pada situs http://www.igoweb.org/~wms/comp/cgoban/index.html. Game tersebut dapat dilihat pada gambar 6. 4



Gambar 6.4 Game CGoban

BAB VII

KESIMPULAN DAN SARAN

7.1 Kesimpulan

Setelah dilakukan pengujian dapat ditarik kesimpulan sebagai berikut :

1. Program permainan Othello ini telah dapat berjalan sesuai dengan yang direncanakan, yaitu system telah mampu melakukan koneksi antar jaringan sehingga permainan dapat dijalankan dengan menggunakan dua computer.
2. Dengan menggunakan OpenGL, tampilan pada permainan terasa lebih baik.

7.2 Saran

Beberapa saran untuk pengembangan dan penelitian selanjutnya sebagai berikut :

1. Tidak adanya level permainan dalam perangkat lunak permainan Othello ini diharapkan ada pengembangan lebih lanjut dengan tingkatan bermain yang lebih bervariasi. Seperti level untuk pemula, menengah dan mahir.
2. Perangkat lunak permainan othello ini sudah mendukung untuk permainan dua komputer dengan menggunakan jaringan, akan tetapi jaringan yang digunakan masih cukup sederhana. Dengan ini diharapkan adanya pengembangan program dengan menggunakan sistem yang lebih baik dan sempurna lagi sehingga mampu mendukungpermainan yang berbasis web

dengan menggunakan jaringan komputer , sehingga *user* dapat langsung memainkan Othello ini melalui suatu *website*.

3. Visualisasi dalam perangkat lunak permainan Othello ini sudah menambahkan animasi 3D dalm proses buah-buah yang akan dibalik dalam permainan, akan tetapi animasi ini masih tergolong "biasa", sehingga diharapkan pada pengembangan permainan ini di masa yang akan datang akan menampilkan animasi 3D yang lebih bagus, hal ini yang akan membuat permainan terasa lebih menarik dan menyenangkan.

DAFTAR PUSTAKA

[AHO00] *Anonimous, A History of Othello,* http//www.othello.org.hk di akses Oktober 2006.

[ASH01] A. Suhendar dan Hariman Gunadi, *Visual Modelling Menggunakan UML dan Rational Rose,* Bandung : Informatika Bandung, 2002.

[BEN02] Bennett, S., McRobb, S., dan Farmer, R. *Object-Oriented Systems Analysis And Design Using UML, second edition.* United Kingdom: Mc Graw-Hill International, 2002.

[DHA03] Dharmiyanti Sri., *Pengantar Unified Modelling Language (UML).http//www.ilmukomputer.com, kuliahumum,* diakses Januari 2007.

[HEN89] Hendra, *Membuat program permainan.* Jakarta : Elex Media Komputindo. 1989.

[MOD05] *Anonimous,* www.modelmakertools.com/uml-explorer/ screenshoots/page1-page4.html, di akses tanggal 30 Mei 2007.