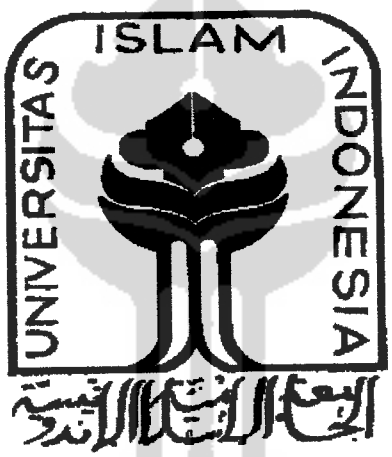
PEMBUATAN APLIKASI PERMAINAN ULAR TANGGA ANTAR MOBILE DEVICE MENGGUNAKAN KONEKSI BLUETOOTH DENGAN J2ME

TUGAS AKHIR

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



oleh:

Nama : **Windy Puspita Sari**
No. Mahasiswa : **03 523 041**

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2007

LEMBAR PENGESAHAN PEMBIMBING

PEMBUATAN APLIKASI PERMAINAN ULAR TANGGA ANTAR MOBILE DEVICE MENGGUNAKAN KONEKSI BLUETOOTH DENGAN J2ME

TUGAS AKHIR

Oleh :

Nama : **WINDY PUSPITA SARI**

No. Mahasiswa : **03 523 041**

Yogyakarta, 16 April 2007

Pembimbing

(Taufiq Hidayat ST,MCS)

Halaman Persembahan

I dedicate to :
My Mom & Dad
My only one brother Armono W
My best friend Putri Nazma Maharani

MOTTO

"Sungguh bersama kesukaran pasti ada kemudahan" (QS. Al-Insyirah : 5)

"dan bersama kesukaran pasti ada kemudahan" (QS. Al-Insyirah : 6)

"karena itu, bila selesai suatu tugas, mulailah tugas yang lain dengan sungguh-sungguh" (QS. Al-Insyirah : 7)

"Hanya kepada Tuhanmulah hendaknya kamu berharap" (QS. Al-Insyirah : 8)

"Jadilah diri sendiri, jangan pernah berusaha menjadi orang lain"

"Kerjakanlah sesuatu yang dapat dikerjakan terlrbih dahulu"

"Jangan pernah menyerah sebelum mencoba dan berusaha"

"Jangan selalu mengharapkan bantuan orang lain"

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Alhamdulillah, segala puji dan syukur penulis panjatkan kehadirat Allah SWT yang telah melimpahkan rahmat, taufik serta hidayahnya. Sholawat dan salam kepada junjungan kita Nabi Muhammad SAW beserta keluarga dan para sahabat, serta orang-orang yang bertaqwa, sehingga penulis dapat menyelesaikan Tugas Akhir ini sebagaimana mestinya.

Tugas Akhir ini merupakan salah satu penerapan ilmu yang telah didapatkan selama kuliah. Dengan adanya penelitian ini, penyusun InsyaAllah akan dapat memahami penggunaan koneksi *bluetooth* dalam aplikasi permainan ular tangga dengan J2ME.

Penyusun menyampaikan ucapan terimakasih dan penghargaan yang setinggi-tingginya atas bantuan, bimbingan dan dukungan dari berbagai pihak yang ikut serta demi kelancaran pelaksanaan Tugas Akhir kepada :

1. Bapak Fathul Wahid ST, MSc selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
2. Bapak Yudi Prayudi S.Si, M.Kom selaku Ketua Jurusan Teknik Informatika Universitas Islam Indonesia.
3. Bapak Taufiq Hidayat ST, MCS selaku dosen pembimbing. Terima kasih atas segala bantuan, dukungan, semangat, dan pengetahuannya, serta kemudahan yang telah diberikan.

4. Kedua orang tua penyusun atas kasih sayang dan perhatian yang telah diberikan selama ini.
5. Buat ***Mas Mono*** atas dukungan, semangat dan bimbingannya.
6. Buat Putri, Mas Edo, Mas Teddy dan anak-anak **'Sailor'** (Al, Ken, Sasty dan Tika) terima kasih atas semangat, dorongan dan waktunya yang telah diberikan selama penyusun mengerjakan tugas akhir ini.
7. Buat Mas Fafan, Diko, Mas Ichal, Mas Fauzan dan Mas Andan yang telah berbaik hati memberikan referensi untuk mengerjakan tugas akhir ini.
8. Teman-teman **Lab Informatika Terpadu,** terimakasih atas kekompakan dan kebersamannya selama ini.
9. Semua pihak yang tidak dapat penyusun sebutkan satu persatu yang telah membantu sejak pengumpulan data sampai penyusunan Tugas Akhir ini.

Semoga amal ibadah dan kebaikan yang telah diberikan mendapatkan imbalan yang setimpal dari Allah SWT.

Penyusun menyadari bahwa Tugas Akhir ini masih jauh dari kesempurnaan, untuk itu penyusun sangat mengharapkan kritik serta saran yang bersifat membangun untuk perbaikan di masa mendatang. Semoga Tugas Akhir ini bermanfaat untuk kita semua. Amin.

Wassalum'alaikum Wr. Wb.

Yogyakarta, 15 April 2007

penyusun

DAFTAR ISI

LEMBAR PENGESAHAN PEMBIMBING i

LEMBAR PENGESAHAN PENGUJI ii

HALAMAN PERSEMBAHAN iii

HALAMAN MOTTO v

KATA PENGANTAR vi

DAFTAR ISI viii

DAFTAR GAMBAR xii

DAFTAR TABEL xv

ABSTRAKSI xvi

I. BAB I PENDAHULUAN

1.1 Latar Belakang Masalah 1

1.2 Rumusan Masalah 1

1.3 Batasan Masalah 2

1.4 Tujuan Penelitian 2

1.5 Manfaat Penelitian 3

1.6 Metodologi Penelitian 3

1.6.1 Metode Pengumpulan Data 3

1.6.1 Metode Pengembangan Software 3

1.7 Sistematika Penulisan 4

II. BAB II LANDASAN TEORI

2.1 Tinjauan Pustaka 6

2.2 Teori Dasar 7

2.2.1 J2ME 7
2.2.1.1 Konfigurasi J2ME 7
2.2.1.2 Profil J2ME 10
2.2.2 Bluetooth 12
2.2.3 Java APIs for Bluetooth 13
2.2.4 Permainan Ular Tangga 14
2.2.5 Konsep Permainan pada Jaringan 16

III. BAB III METODOLOGI

3.1 Analisis Kebutuhan Perangkat Lunak 19
3.1.1 Metode Analisis 19
3.1.2 Hasil Analisis 19
3.1.2.1 Analisis kebutuhan Input 20
3.1.2.2 Analisis kebutuhan Output 20
3.1.2.3 Analisis kebutuhan Fungsi 21
3.1.2.4 Analisis kebutuhan Perangkat Keras 21
3.1.2.5 Analisis kebutuhan Perangkat Lunak 22
3.1.2.6 Analisis kebutuhan Antar Muka 22
3.2 Perancangan Perangkat Lunak 23
3.2.1 Metode Perancangan 23
3.2.2 Hasil Perancangan 23
3.2.2.1 Use Case 23
3.2.2.2 Class Diagram 25
3.2.2.3 Sequence Diagram 32
3.2.2.3.1 Sequence Diagram untuk Skenario Mulai Bermain Satu Player 32
3.2.2.3.2 Sequence Diagram untuk Skenario Mulai Bermain Dua Player dan Menjadi Server 33

3.2.2.3.3 Sequence Diagram untuk Skenario Mulai Bermain Dua Player dan Menjadi Client 33
3.2.2.3.4 Sequence Diagram untuk Skenario Pengaturan Suara 34
3.2.2.3.5 Sequence Diagram untuk Skenario Memilih Skin Papan Permainan 35
3.2.2.3.6 Sequence Diagram untuk Skenario Melihat Bantuan Permainan 36
3.2.2.3.7 Sequence Diagram untuk Skenario Melihat Informasi Pembuat 36
3.2.3 Perancangan Antar Muka (Interface) 37
3.2.3.1 Rancangan Antar Muka Menu Utama 37
3.2.3.2 Rancangan Antar Muka Menu Start Game 38
3.2.3.3 Rancangan Antar Muka Menu Sound 38
3.2.3.4 Rancangan Antar Muka Menu Skins 39
3.2.3.5 Rancangan Antar Muka Halaman Help 40
3.2.3.6 Rancangan Antar Muka Halaman About 40
3.2.3.7 Rancangan Antar Muka Menu 2 Players 41
3.2.3.8 Rancangan Antar Muka Halaman Set Volume 42
3.2.3.9 Rancangan Antar Papan Permainan 42
3.3 Implementasi Perangkat Lunak 43
3.3.1 Batasan Implementasi 44
3.3.2 Implementasi antarmuka 44
3.3.2.1 Halaman Splash Screen 44
3.3.2.2 Menu Utama 45
3.3.2.3 Menu Start Game 46
3.3.2.4 Menu Sound 46
3.3.2.5 Menu Skins 47

3.3.2.6 Halaman Help..... 48
3.3.2.7 Halaman About..... 48
3.3.2.8 Menu 2 Players..... 49
3.3.2.9 Halaman Set Volume..... 50
3.3.2.10 Papan Permainan..... 50
3.3.3 Implementasi prosedural..... 51
3.3.3.1 Fungsi untuk Mengocok Dadu..... 52
3.3.3.2 Fungsi untuk Menghentikan Kocokan Dadu..... 52
3.3.3.3 Fungsi untuk Menjalankan Bidak (Pion)..... 52
3.3.3.4 Fungsi untuk Mengecek Apakah Bidak (Pion) Bertemu Ular atau Tangga..... 53
3.3.3.5 Fungsi untuk Menjalankan Bidak (Pion) Mengikuti Tangga atau Ular..... 54
3.3.3.6 Fungsi Jalan Mundur pada Bidak (Pion)..... 54
3.3.3.7 Fungsi untuk Mengecek Apakah Bidak (Pion) Telah Memenangkan Permainan..... 55

IV. BAB IV HASIL DAN PEMBAHASAN

4.1 Pengujian Program..... 56
4.2 Analisis Kinerja Sistem..... 56
4.2.1 Penanganan Kesalahan..... 56
4.2.2 Pengujian dan Analisis..... 57

V. BAB V PENUTUP

5.1 Kesimpulan..... 64
5.2 Saran..... 64

DAFTAR PUSTAKA

DAFTAR GAMBAR

Gambar 2.1	Lingkungan kerja teknologi Java	8
Gambar 2.2	Scatternet	13
Gambar 2.3	Bluetooth dan J2ME IDP	14
Gambar 2.4	Contoh papan permainan ular tangga	15
Gambar 3.1	Use Case Diagram	25
Gambar 3.2	Class Diagram	31
Gambar 3.3	Sequence diagram untuk skenario mulai bermain satu player	33
Gambar 3.4	Sequence diagram untuk skenario mulai bermain dua player dan menjadi server	33
Gambar 3.5	Sequence diagram untuk skenario mulai bermain dua player dan menjadi client	34
Gambar 3.6	Sequence diagram untuk skenario pengaturan suara	35
Gambar 3.7	Sequence diagram untuk skenario memilih skin papan permainan	35
Gambar 3.8	Sequence diagram untuk skenario melihat bantuan permainan	36
Gambar 3.9	Sequence diagram untuk skenario melihat informasi pembuat	36
Gambar 3.10	Menu Utama	37
Gambar 3.11	Menu Start Game	38
Gambar 3.12	Menu Sound	39
Gambar 3.13	Menu Skins	39
Gambar 3.14	Halaman Help	40

Gambar 3.15 Halaman About 41
Gambar 3.16 Menu 2 Players 41
Gambar 3.17 Halaman Set Volume 42
Gambar 3.18 Papan Permainan 43
Gambar 3.19 Tampilan Splash Screen 45
Gambar 3.20 Tampilan Menu Utama 45
Gambar 3.21 Tampilan Menu Start Game 46
Gambar 3.22 Tampilan Menu Sound 47
Gambar 3.23 Tampilan Menu Skins 47
Gambar 3.24 Tampilan Halaman Help 48
Gambar 3.25 Tampilan Halaman About 49
Gambar 3.26 Tampilan Menu 2 Players 49
Gambar 3.27 Tampilan Halaman Set Volume 50
Gambar 3.28 Tampilan Papan Permainan 51
Gambar 4.1 Tampilan pesan *error* bila bluetooth belum dinyalakan 57
Gambar 4.2 Antar muka menu Utama ketika *keypad down* ditekan lima kali .. 58
Gambar 4.3 Antar muka menu Utama ketika *keypad center* kemudian ditekan 58
Gambar 4.4 Antar muka menu *Start Game* ketika *keypad up* ditekan tiga kali 59

Gambar 4.5 Antar muka menu *Sound* ketika *keypad up* ditekan satu kali lalu *keypad center* ditekan..... 59
Gambar 4.6 Antar muka menu *Skins* ketika *keypad down* ditekan tiga kali lalu *keypad center* ditekan..... 60
Gambar 4.7 Antar muka menu 2 *Players* ketika *keypad down* ditekan satu kali..... 61
Gambar 4.8 Antar muka halaman *Set Volume* ketika *keypad left* ditekan tiga kali..... 61
Gambar 4.9 Antar muka Papan Permainan ketika bidak dijalankan lima langkah..... 62
Gambar 4.10 Antar muka halaman Konfirmasi Melanjutkan Permainan ketika *keypad down* ditekan satu kali..... 63

DAFTAR TABEL

Tabel 2.1 Perbandingan CLDC dan CDC 10

ABSTRAKSI

Di tengah kemajuan teknologi yang pesat, manusia modern dituntut untuk selalu *mobile*. Memainkan suatu permainan dari *mobile device* juga biasa dilakukan. Namun menjadi tidak menyenangkan bila harus memainkannya dengan pemain lainnya dari *mobile device* yang sama. Untuk itu perlu adanya suatu cara yang memungkinkan suatu permainan dimainkan bersama-sama tetapi dari *mobile device* masing-masing menggunakan koneksi nirkabel. Dan koneksi nirkabel yang saat ini banyak digunakan adalah *bluetooth*. Tujuan dari penelitian ini adalah membuat aplikasi dengan J2ME yang memanfaatkan *bluetooth* untuk digunakan dalam permainan *multi player* di *mobile device*. Sehingga memudahkan *user* untuk memainkan permainan *multi player* tersebut. Permainan yang diimplementasikan dalam penelitian ini adalah ular tangga.

Hasil dari penelitian ini adalah aplikasi permainan ular tangga yang dapat dimainkan *single player* atau pun *multi player* (maksimal dua *player*) dengan menggunakan koneksi *bluetooth*.

Kata kunci : *Bluetooth*, J2ME, *Games*, Ular Tangga

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Di tengah kemajuan teknologi yang pesat, manusia modern dituntut untuk selalu *mobile*. Misalnya melakukan transaksi *banking via mobile*, membaca Al-qur'an dari *mobile* qur'an, dan sebagainya. Cara ini dianggap efisien dan praktis, karena dapat dilakukan hanya dari genggaman *mobile device*.

Memainkan suatu permainan dari *mobile device* juga biasa dilakukan. Namun menjadi tidak menyenangkan bila harus memainkannya bersama-sama dengan pemain lainnya dari *mobile device* yang sama, terutama untuk permainan yang *multi player*. Untuk itu perlu adanya suatu cara yang memungkinkan suatu permainan dimainkan bersama-sama tetapi dari *mobile device* masing-masing menggunakan koneksi nirkabel.

Dua koneksi nirkabel yang disediakan oleh *mobile device* adalah *infrared* dan *bluetooth*. Dan koneksi yang saat ini banyak digunakan adalah *bluetooth*, karena memiliki jangkauan yang lebih jauh dibandingkan *infrared*.

1.2 Rumusan Masalah

Bagaimana merancang dan membangun suatu aplikasi J2ME yang bisa diterapkan pada suatu permainan, agar dua orang pemain dapat memainkannya dari *mobile device* masing-masing menggunakan koneksi *bluetooth*.

1.3 Batasan Masalah

Agar lebih mudah dalam penarikan kesimpulan serta menjaga agar lebih mencerminkan permasalahan yang sedang dihadapi, maka diperlukan batasan masalah sebagai berikut :

a. Aplikasi hanya dapat digunakan untuk maksimal dua *mobile device* (smartphone/PDA).
b. *Mobile device* yang digunakan harus java *enable* dengan MIDP 2.0 dan memiliki konektifitas *bluetooth*.
c. *Mobile device* yang digunakan minimal memiliki ukuran layar 176 x 208 pixel.
d. *Mobile device* yang digunakan harus mempunyai Java APIs for *Bluetooth* Wireless Technology (JABWT / JSR-82).
e. Dalam penelitian ini, permainan yang diimplementasikan adalah ular tangga.
f. Aplikasi tidak menyimpan skor dari permainan karena *mobile device* yang digunakan oleh lawan main belum tentu sama.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah sebuah aplikasi menggunakan J2ME yang:

a. Memanfaatkan *bluetooth* untuk digunakan dalam permainan *multi player* di *mobile device*.
b. Mempermudah *user* dalam memainkan suatu permainan secara bersama-sama dari *mobile device* masing-masing.

1.5 Manfaat Penelitian

Adapun manfaat yang diperoleh dari penelitian ini adalah :

a. Mempermudah *user* dalam memainkan permainan *multi player* di *mobile device*.

b. Meningkatkan fungsi *bluetooth* sebagai koneksi nirkabel pada *mobile device*.

1.6 Metodologi Penelitian

Adapun metode-metode yang akan digunakan dalam penyelesaian tugas akhir ini adalah sebagai berikut :

a. Metode Pengumpulan Data

b. Metode Pengembangan Software

1.6.1 Metode Pengumpulan Data

Metode pengumpulan data adalah metode yang digunakan untuk mengumpulkan data yang diperlukan dalam penelitian. Metode yang digunakan adalah metode *Library Research*, yaitu pengumpulan data dengan cara melakukan studi, analisis dan dokumentasi literatur, dan sumber catatan lain yang berkaitan dengan permasalahan yang dibahas.

1.6.2 Metode Pengembangan Software

Pada penelitian ini metode pengembangan yang digunakan adalah metode waterfall yang merupakan metode pengembangan perangkat lunak yang sistematis. Pendekatan sekuensial dimulai dari level sistem, kemudian analisis,

design, *coding*, *testing* dan *maintenance*. Analisis kebutuhan perangkat lunak meliputi analisis kebutuhan masukan data, analisis kebutuhan keluaran data, analisis kebutuhan sistem dan analisis kebutuhan antarmuka.

1.7 Sistematika Penulisan

Untuk memudahkan dalam memahami laporan Tugas Akhir, dikemukakan sistematika penulisan yang terdiri dari 5 bab, yaitu:

BAB I berisi pengantar terhadap permasalahan yang akan dibahas. Di dalamnya menguraikan tentang gambaran suatu penelitian yang terdiri dari : latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan.

BAB II berisi mengenai teori–teori yang berhubungan dengan teknologi Java 2 Micro Edition (J2ME) , *Bluetooth*, permainan ular tangga dan konsep permainan pada jaringan yang menjadi landasan dalam melakukan penelitian.

BAB III berisi tentang uraian langkah-langkah penyelesaian masalah, yaitu metode analisis yang dipakai pada analisis kebutuhan perangkat lunak, metode perancangan *Aplikasi Permainan Ular Tangga Menggunakan Koneksi Bluetooth dengan J2ME* serta batasan implementasi dan dokumentasinya.

BAB IV berisi uraian tentang hasil pengujian *Aplikasi Permainan Ular Tangga Menggunakan Koneksi Bluetooth dengan J2ME* dan bagaimana hasil tersebut dicapai, serta pembahasannya.

BAB V berisi kesimpulan yang diperoleh dari pemecahan masalah maupun penelitian serta saran-saran sebagai masukan untuk perbaikan di masa yang akan datang.

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Pada tugas akhir yang berjudul "MEMBANGUN PERMAINAN ULAR TANGGA BERBASISKAN MULTIMEDIA" yang disusun oleh Ahmadi Yuli Ananta (99523017), permainan ular tangga yang dibuat adalah jenis *game desktop*. Dalam permainan tersebut pemain dapat memilih jumlah pemain dua sampai empat orang. Pemain juga dapat menentukan jumlah kotak dari papan permainan (8x8, 10x10, atau 15x15 kotak). Posisi ular dan tangga pun dibuat *random* agar pemain tidak bosan memainkannya [ANA05].

Meskipun pemain dapat berjumlah dua sampai empat orang, namun permainan ini harus dimainkan dalam satu komputer yang sama, sehingga pemain harus berganti-gantian menggunakan komputernya. Tentu saja akan lebih nyaman bila dapat memainkannya di *device* masing-masing.

Tugas akhir yang mengangkat tentang permainan berbasis *bluetooth*, yaitu yang berjudul "RANCANG BANGUN APLIKASI GAME JARINGAN PADA PONSEL BERBASIS BLUETOOTH MENGGUNAKAN TEKNOLOGI MIDP 2.0 DAN CLDC 1.1" . Tugas akhir yang disusun oleh Dwie Sevinna Apriany (00523230) ini, mengimplementasikan permainan Dam–Dam-An. Aplikasi yang dibuat telah dilengkapi dengan efek suara dan fungsi pause *game* serta sudah *full colour* [APR06].

Aplikasi tersebut harus dimainkan oleh dua orang, karena tidak ada fasilitas untuk dapat memainkannya melawan komputer. Sehingga pemain harus mencari lawan mainnya terlebih dahulu yang juga memiliki *device* yang *compatible* serta telah terinstal permainan Dam-Dam An.

2.2 Teori Dasar

2.2.1 J2ME

J2ME merupakan sebuah versi yang direduksi dari Java API dan *Java Virtual Machine* (JVM) yang didesain untuk dapat dioperasikan dalam komputer dan mikrokomputer. J2ME diarahkan untuk diaplikasikan pada piranti komputasi kecil seperti ponsel, PDA, palm, pager, dll. Untuk mendukung berbagai macam piranti ini, maka J2ME diorganisasikan ke dalam konfigurasi dan profil. Baik profil maupun konfigurasi mendefinisikan sebuah kelas Java API yang dapat digunakan oleh aplikasi [SUY05].

J2ME dalam portabilitas kemampuan dapat dijalankan dimana pun dan *safe network delivery* seperti J2SE dan J2EE. J2ME menggunakan sekumpulan paket kecil. J2ME berisi subset dari paket-paket di J2SE ditambah paket spesifik Micro Edition berupa *Javax.microedition.io.* Aplikasi-aplikasi J2ME dapat diskala agar dapat bekerja dengan J2SE dan J2EE [WIC02].

2.2.1.1 Konfigurasi J2ME

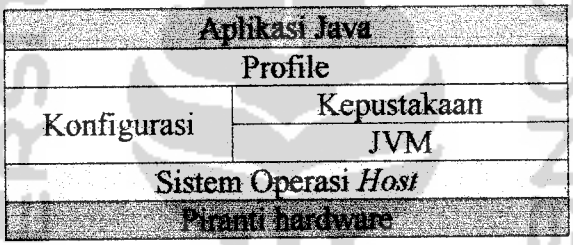
Untuk mendukung berbagai jenis produk ponsel dan PDA yang sesuai dengan *scope* J2ME, Sun memperkenalkan konfigurasi. Konfigurasi tersebut mempunyai tiga elemen dasar yaitu:

a. Sebuah set fitur bahasa pemrograman Java.

b. Sebuah set fitur *Java Virtual Machine* (JVM).

c. Sebuah set kepustakaan pendukung kepustakaan Java dan *application programming interfaces* (API).

Konfigurasi J2ME ditunjukkan pada Gambar 2.1.



Gambar 2.1 lingkungan kerja teknologi Java

Dengan konfigurasi dapat ditentukan beberapa hal yang perlu diperhatikan dalam pengembangan aplikasi yang berjalan di *mobile device* (ponsel, PDA, dll), misalnya memori, *display*, konektifitas jaringan dan kemampuan pemrosesannya [SUY05].

Konfigurasi ditentukan perkembangannya oleh JCP *(Java Community Process)*, inilah badan non-profit yang berkutat dengan perkembangan teknologi Java. Saat ini telah didefenisikan dua buah konfigurasi yaitu: CLDC (*Connected Limited Device Configuration)* dan CDC (*Connected Device Configuration)*.

1. CLDC (*Connected Limited Device Configuration)*

CLDC menyediakan *platform* Java standar yang cocok untuk perangkat kecil dengan sumber daya terbatas. Perangkat ini umumnya mempunyai pemrosesan 16 *bit* atau 32 *bit* dan memori 160 *Kbyte* sampai 512 *Kbyte.* Perangkat-perangkat ini diberdayakan dengan baterai dan mempunyai konektifitas suatu jenis jaringan, biasanya *wireless* dengan koneksi tidak tetap ber*bandwith* 9600 bps, layar tampilan sempit. Inti CLDC adalah *Kilobyte Virtual Machine* (KVM).

2. CDC (*Connected Device Configuration)*

CDC menyediakan mesin maya dan pustaka kelas dasar yang mendukung aplikasi Java untuk perangkat seperti *smart communicator, pager,* PDA *(Personal Digital Asistant),* dan *television set-top Box.* Perangkat-perangkat ini umumnya memiliki pemrosesan 32 bit dan memori lebih dari 2 MB.

Konfigurasi ini memiliki kemampuan untuk melakukan koneksi internet. Fungsionalitas mesin maya Java dipenuhi oleh CVM *virtual machine* yang merupakan mesin maya dengan fitur penuh. Tabel 2.1 adalah perbandingan antara CDC dan CLDC [WIC02].

Tabel 2.1 Perbandingan CLDC dan CDC

CLDC	CDC
Mengimplementasikan subset dari J2SE	Mengimplementasikan seluruh fitur pada J2SE
JVM yang digunakan dikenal dengan KVM	JVM yang digunakan dikenal dengan CVM
Digunakan pada perangkat *handheld* dengan ukuran memori terbatas (160 – 512 Kb)	Digunakan pada perangkat *handheld* dengan ukuran memori minimal 2 Mb
Prosesor 16 bit atau 32 bit	Prosesor 32 bit

Walupun pembagian konfigurasi J2ME (CLDC dan CDC) nampak sangat jelas, namun sekarang ini sudah banyak *mobile device* yang mempunyai spesifikasi yang sangat hebat sehingga pembagian menurut memori dan prosesor nampak tidak relevan lagi. Untuk itu, perbedaan CLDC dan CDC seperti ditunjukkan pada tabel 2.1 nampaknya perlu diperbaiki lagi. CLDC adalah bagian dari CDC, baik CLDC maupun CDC merupakan bagian dari platform J2SE [SUY05].

2.2.1.2 Profil J2ME

Profile adalah spesifikasi yang merinci kumpulan API (*Application Programming Interface)* yang dibangun menggunakan konfigurasi. Salah satu contoh terkenal profile adalah *Foundation Profile* yang dibangun pada CDC. Profile ini menyediakan lingkungan jalan J2ME lengkap untuk aplikasi yang ditargetkan pada *gateway, smartphone,* dan pager dua arah.

J2ME profile menyediakan implementasi tambahan yang sangat spesifik dari sebuah handheld device. Ada 5 kategori J2ME Profile, yaitu :

1. **MIDP (*Mobile Information Device Profile)***

   MIDP menyediakan *library-library* Java untuk implementasi dasar antarmuka (GUI), implementasi jaringan *(networking)*, database dan timer. MIDP dirancang khusus untuk *wireless phone* dan pager. MIDP ditargetkan untuk perangkat komunikasi dua arah yang mengimplementasikan J2ME CLDC, dengan fasilitas *toolkit* tampilan, metode pemasukan dari pemakai, penyimpanan data menggunakan model basisdata berorientasi record sederhana, jaringan berbasis HTTP menggunakan CLDC *generic connection framework*. MIDP di desain untuk menambah fungsionalitas dari spesifikasi CLDC.

   Spesifikasi MIDP meliputi :

   a. Dukungan *user-interface (Limited Connected Device User Interface*, LCDUI).
   b. Dukungan jaringan (basis protocol HTTP dan *Generic Connection Frame Work* pada CLDC).
   c. Dukungan *persistent storage (Record Management System*, RMS).
   d. Class dukungan yang lain, seperti *timer* dan *exceptions.*

2. **Foundation Profile**

   Profile dasar untuk non-GUI network devices pada CDC.

3. **Personal Profile**

   Profil yang menyediakan dukungan penuh AWT dan *compatible* dengan J2SE versi 1.3.1.

4. **PDA (*Personal Digital Assistance)* Profile**

   Profil yang menggunakan konfigurasi CLDC.

5. **RMI Profile**

   Profil yang dirancang untuk platform yang mendukung konfigurasi CDC. Profil ini membutuhkan implementasi dari *Foundation Profile.*

MIDP ditargetkan untuk perangkat komunikasi dua arah yang mengimplementasikan J2ME CLDC, dengan fasilitas *toolkit* tampilan, metode pemasukan dari pemakai, penyimpanan data menggunakan model basis data berorientasi record sederhana, jaringan berbasis HTTP menggunakan CLDC *generic connection framework* [SUY05].

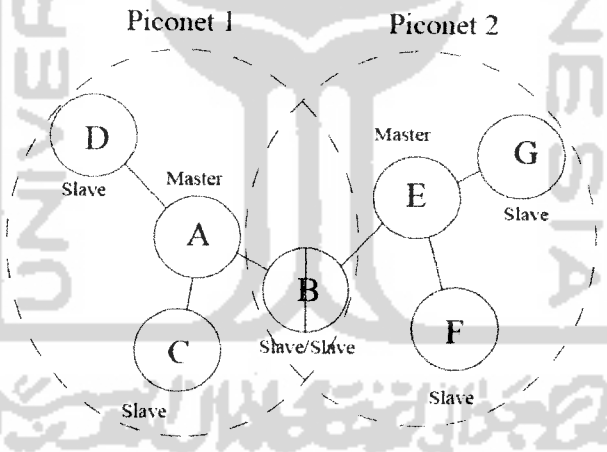
Pada pertengahan bulan November 2003, muncul MIDP versi 2.0. Di samping masih mengandalkan beberapa fitur lama yang cukup stabil misalnya penanganan *user interface* dan *record management*, dalam versi ini beberapa fitur lama diperbaharui dan dilengkapi. Tiga fitur penting yang terdapat pada MIDP 2.0 ini adalah: Multimedia, *Game* API, dan *Secure Networking* [HAR04].

2.2.2 Bluetooth

Bluetooth adalah teknologi radio jarak pendek yang memungkinkan koneksi *wireless* antar *mobile device*. Tiga tujuan utama dari *bluetooth* adalah: ukuran kecil, pemakaian tenaga minimal, dan harga rendah.

Jaringan *bluetooth* disebut *piconet*. Contoh yang paling sederhana dari jaringan buetooth adalah bila dua *device* terkoneksi (saling berhubungan). *Device* yang memulai koneksi disebut *master* dan *device* lainnya disebut *slave*. Koneksi *bluetooth* adalah khusus koneksi *ad hoc*, yang berarti bahwa jaringan akan dibangun hanya untuk tugas saat itu saja dan kemudian akan diputus jaringannya setelah transfer data selesai.

Master dapat mempunyai koneksi bersama (*point to multipoint*) sampai tujuh *slave*. Lalu, bagaimanapun, kecepatan data terbatas. Sebuah *device* dapat juga terkoneksi pada dua atau lebih *piconet* yang disebut *scatternet*. Sebuah *device* bagaimanapun, hanya dapat menjadi *master* pada satu *piconet* di satu waktu. Bentuk terjadinya *scatternet* dapat dilihat pada Gambar 2.2 [XXX03].

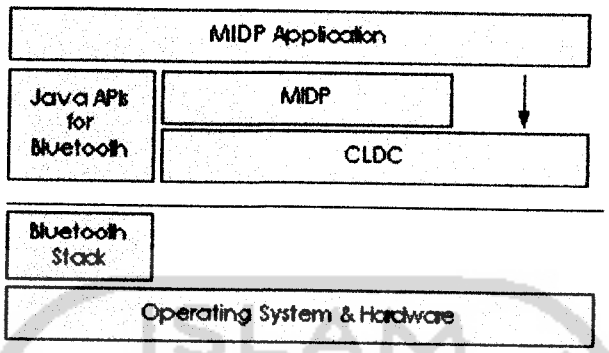


Gambar 2.2 *Scatternet*

2.2.3 Java APIs *for Bluetooth*

Java APIs *for bluetooth* adalah *optional package* J2ME yang dimaksudkan untuk memberikan definisi *bluetooth* oleh *Java Community Process* (JSR-82). *Optional package* ini menyediakan pengembangan API untuk *bluetooth* yang

umum. Gambar 2.3 memberikan ilustrasi tentang hubungan antara Java APIs *for bluetooth* dan J2ME *platform*, menggunakan MIDP dan CLDC *stack* [XXX04].

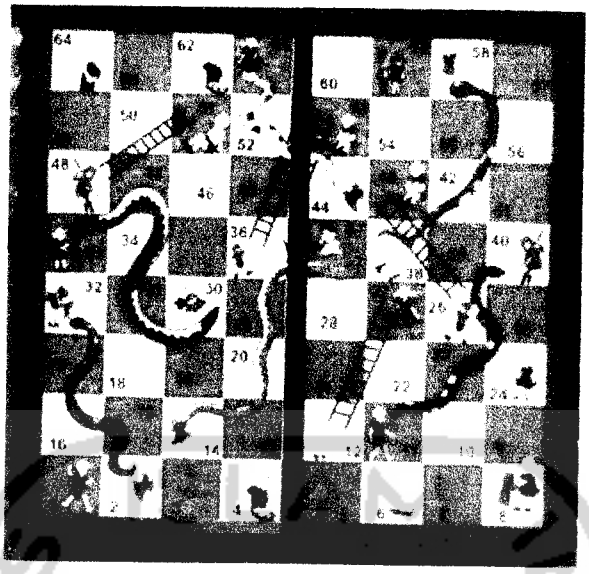


Gambar 2.3 *Bluetooth dan J2ME MIDP*

2.2.4 Permainan Ular Tangga

Ular tangga adalah permainan papan untuk anak-anak yang dimainkan oleh 2 orang atau lebih. Papan permainan dibagi dalam kotak-kotak kecil dan di beberapa kotak digambar sejumlah "tangga" atau "ular" yang menghubungkannya dengan kotak lain. Permainan ini diciptakan pada tahun 1870.

Tidak ada papan permainan standar dalam ular tangga - setiap orang dapat menciptakan papan mereka sendiri dengan jumlah kotak, ular dan tangga yang berlainan. Contoh papan permainan ular tangga dapat dilihat pada gambar 2.4.



Gambar 2.4 contoh papan permainan ular tangga

Setiap pemain mulai dengan bidaknya di kotak pertama (biasanya kotak di sudut kiri bawah) dan secara bergiliran melemparkan dadu. Bidak dijalankan sesuai dengan jumlah mata dadu yang muncul. Bila pemain mendarat di ujung bawah sebuah tangga, mereka dapat langsung pergi ke ujung tangga yang lain. Bila mendarat di kotak dengan ular, mereka harus turun ke kotak di ujung bawah ular. Misalnya dilihat pada gambar 2.4, bila pemain mendarat di kotak 12, maka pemain dapat langsung pergi ke kotak 27. Namun bila pemain mendarat di kotak 32, maka pemain harus turun ke kotak 2. Pemenang adalah pemain pertama yang mencapai kotak terakhir [XXX07].

2.2.5 Konsep Permainan pada Jaringan

Pada permaianan yang menggunakan jaringan *peer to peer*, frekuensi *update* data yang dikirimkan antar pemain adalah aspek yang sangat penting. Frekuensi update data sangat dipengaruhi oleh interaksi antar *user* dan sinkronisasi yang digunakan [WAN05].

Interaksi antar *user* dapat dibagi menjadi beberapa katagori berikut:

1. ***Controlled*:** Di kategori ini, interaksi pemain dalam permainan dikendalikan melalui sebuah *well-defined protocol*, dimana salah satu pemain di dalam jaringan tersebut harus menjadi *master controlling* dari interaksi pemain. Untuk permainan di kategori ini, waktu berinteraksi tidak ditentukan oleh pemain, sejak *master* mengontrol interaksi. Kategori ini cocok dengan *game* yang berdasarkan *turn-based* atau *game* yang pemainnya harus berinteraksi dengan pola tertentu atau urutan tertentu. Kategori ini mencangkup juga permainan-permainan dimana semua pemain harus mengikuti pola atau rute tertentu (contoh: *driving games*).

2. ***User interaction*:** Di kategori ini, pemain harus secara gamblang memicu tindakan yang akan menyebabkan interaksi (pertukaran data) dengan pemain lainnya. Kategori ini cocok bagi permainan yang ada jual belinya, misalnya jual beli senjata, perlengkapan dan barang lain di antara pemain. Kategori ini mencakup juga permainan-permainan yang tergantung pemainnya seberapa ingin berinteraksi dengan pemain lainnya.

3. ***Automatic triggered*:** Di kategori ini, *mobile device* milik pemain mencari pemain lain di sekitarnya, dan jika telah menemukan satu pemain dapat memicu sebuah aksi untuk mendapatkan perhatian pemain tersebut (suara atau getaran). Sebagai contoh seperti fungsi di atas dimasukkan pada permainan Nintendogs untuk Nintendo DS. Permainan ini memiliki sebuah *bark mode* dimana anjing virtual milik pemain pada permaianan tersebut akan mulai menggonggong jika pemain lainnya juga berada di *bark mode* pada *wireless range*. Para pemain kemudian dapat berinteraksi dan membiarkan anjing virtual milik mereka saling bertemu.

4. ***Automatic*:** Di kategori ini, *device* milik pemain berinteraksi tanpa *user interacting* dengan permainan itu sendiri. Pada kategori ini, pemain umumnya membentuk dan membuat karakter *autonomous* yang dapat berinteraksi dengan karakter pemain lainnya untuk kepentingan pemain (pemilik karakter). Contoh permainannya adalah *role-playing games*, dimana pemain melengkapi dan melatih karakternya, dan pertarungan antar pemain dalam jaringan berjalan tanpa interaksi pemain.

Sinkronisasi dan *update* data antar pemain dibagi menjadi tiga kategori yaitu:

a. ***Asynchronous:*** *Asynchronous update* adalah untuk *game* jaringan dimana *update* antar pemain bukan waktu yang kritis, tetapi dapat dilakukan

kapanpun. Kategori ini sangat cocok untuk permainan yang *slow-paced* dan tidak membutuhkan data yang *fresh* untuk diproses dalam permainan.

b. ***Synchronous*:** Di kategori ini, partisipasi pemain tergantung pada frekuensi *update* data untuk dapat bermain *game*. Lebih lanjut, untuk kategori ini hanya informasi dalam jumlah kecil yang perlu dikirim antar pemain untuk mempertahankan pemain di *state* yang konsisten. Informasi tersebut dapat berupa skor permainan, *lap-times*, ranking, *simple player state* dan sebagainya.

c. ***Real-time*:** Di kategori ini, permainan mengandalkan pada pertukaran data yang padat antar pemain dalam rangka memberikan pemain pengalaman bermain. Pertukaran data antar pemain dapat berupa posisi karakter atau kendaraan pada dunia umumnya atau trek, posisi karakter atau kendaraan, perpindahan karakter atau kendaraan dan sebagainya

BAB III

METODOLOGI

3.1 Analisis Kebutuhan Perangkat Lunak

3.1.1 Metode Analisis

Tahap analisis digunakan untuk mengetahui dan menerjemahkan semua permasalahan serta kebutuhan perangkat lunak dan kebutuhan sistem yang akan dibangun. Oleh karena itu, tahap analisis digunakan untuk mendapatkan data-data yang diperlukan untuk kepentingan pembangunan sistem. Analisis yang digunakan adalah metode analisis berorientasi objek. Metode ini digunakan karena memiliki beberapa kelebihan. Kelebihan metode ini antara lain :

a. Pengelompokan bahan pembangunan program dengan unsur terkacil yang disebut objek. Unsur terkecil ini bisa diproses sehingga menjadi kesatuan lain yang bersifat berlainan pula, menjadi lebih besar atau lebih khusus dengan proses yang lebih mudah. Seperti duplikasi, pemakaian bersama dan yang lain sebagainya.
b. Dapat dilakukan pemakaian ulang kode
c. Pemeliharaannya lebih mudah

3.1.2 Hasil Analisis

Berdasarkan analisis yang telah dilakukan maka dapat diketahui apa saja yang menjadi masukan sistem, keluaran sistem, fungsi atau metode yang

digunakan dalam sistem, kebutuhan perangkat keras, kebutuhan perangkat lunak serta antarmuka sistem yang akan dibuat, sehingga sistem yang dibuat nantinya sesuai dengan apa yang diharapkan.

3.1.2.1 Analisis Kebutuhan *Input*

Pada aplikasi ini *input* yang dimasukkan *user* adalah berupa pilihan-pilihan dari menu yang telah disediakan, yaitu:

a. Pilihan pada menu utama
b. Pilihan pada menu *Start Game*
c. Pilihan pada menu *Sound*
d. Pilihan pada menu *Skins*
e. Pilihan pada halaman *Help*
f. Pilihan pada halaman *About*
g. Pilihan pada menu *2 Players*
h. Pilihan pada halaman *Set Sound*

Selain itu pada saat *user* menekan *keypad* pada *mobile device* juga merupakan *input* dalam aplikasi ini. Misalnya pada saat *user* akan mulai mengocok dadu, dan memberhentikan kocokan dadu.

3.1.2.2 Analisis Kebutuhan *Output*

Output yang diterima oleh *user* adalah berupa hasil dari pilihan pada menu-menu yang disediakan. Selain itu *output* juga berupa papan permainan,

status giliran, nilai angka dadu yang muncul, pergerakan bidak (pion) dan pemain yang menang.

3.1.2.3 Analisis Kebutuhan Fungsi

Fungsi atau *method* yang dibutuhkan dalam aplikasi permainan Ular Tangga ini adalah:

a. Fungsi untuk mengocok dadu.
b. Fungsi untuk menghentikan kocokan dadu.
c. Fungsi untuk menjalankan bidak (pion).
d. Fungsi untuk mengecek apakah bidak (pion) bertemu dengan ular atau tangga.
e. Fungsi jalan mundur pada bidak (pion) yang telah berada pada kotak terakhir, sementara nilai dadu berlebih.
f. Fungsi untuk mengecek apakah bidak (pion) telah memenangkan permainan.

3.1.2.4 Analisis Kebutuhan Perangkat Keras

Perangkat keras yang digunakan dalam pembuatan aplikasi ini memiliki spesifikasi sebagai berikut :

a. *Prosessor* Intel Pentium Centrino 1.73 GHz.
b. RAM 512 MB DDR2.
c. Intel Graphics Media Accelerator 900.
d. Hardisk 60 GB.

3.1.2.5 Analisis Kebutuhan Perangkat Lunak

Perangkat keras komputer tidak berarti tanpa perangkat lunak begitu juga sebaliknya. Jadi perangkat lunak dan perangkat keras saling mendukung satu sama lain.Perangkat keras hanya berfungsi jika diberikan instruksi-intruksi kepadanya. Instruksi-instruksi inilah disebut dengan perangkat lunak. Dalam penelitian ini, perangkat lunak yang digunakan adalah:

a. Sun Java™ SDK.
b. NetBeans IDE 4.0 digunakan sebagai editor.
c. Mobility Pack 4.0 merupakan paket tambahan pada NetBeans IDE 4.0, untuk dapat mengenali tag J2ME.
d. Sun Java™ Wireless Toolkit 2.2 digunakan sebagai emulator J2ME.
e. Macromedia Flash MX 2004 dan Adobe Photoshop CS2 untuk menggambar *images* yang digunakan pada aplikasi.
f. Nokia PC Suite untuk menginstal aplikasi pada Nokia *smartphone.*

3.1.2.6 Analisis Kebutuhan Antar Muka

Perancangan antar muka menggunakan Macromedia Flash MX 2004 dan Adobe Photoshop CS2, karena Flash dapat digunakan untuk menggambar *vector* dengan mudah dan Photoshop memiliki *tools* untuk menyempurnakan gambar sehingga hasil gambarnya lebih maksimal.

3.2 Perancangan Perangkat Lunak

3.2.1 Metode Perancangan

Karena metode analisis yang dipakai berorientasi objek, maka perancangan aplikasi ini menggunakan salah satu metode analisis berorientasi objek yaitu metode *UML (Unified Modelling Language)* karena metode *UML* dapat dengan mudah menjelaskan objek-objek dan kelas-kelas.

3.2.2 Hasil Perancangan

UML memiliki beberapa konsep dasar yang diabstraksikan dalam bentuk *structural classification*, *dynamic behavior*, dan *model management*. Hal terpenting dalam penggunaan UML adalah pembuatan diagram yang sesuai dengan analisis dan pengembangan sistem. Hasil perancangan sistem ini dibedakan menjadi beberapa bagian sesuai dengan tahapan-tahapan yang digunakan pada metode perancangannya. Pada tahap perancangan ini, dibatasi pada pembuatan tiga diagram saja, yaitu :

a. *Use Case*
b. *Class Diagram*
c. *Sequence diagram*

3.2.2.1 *Use Case*

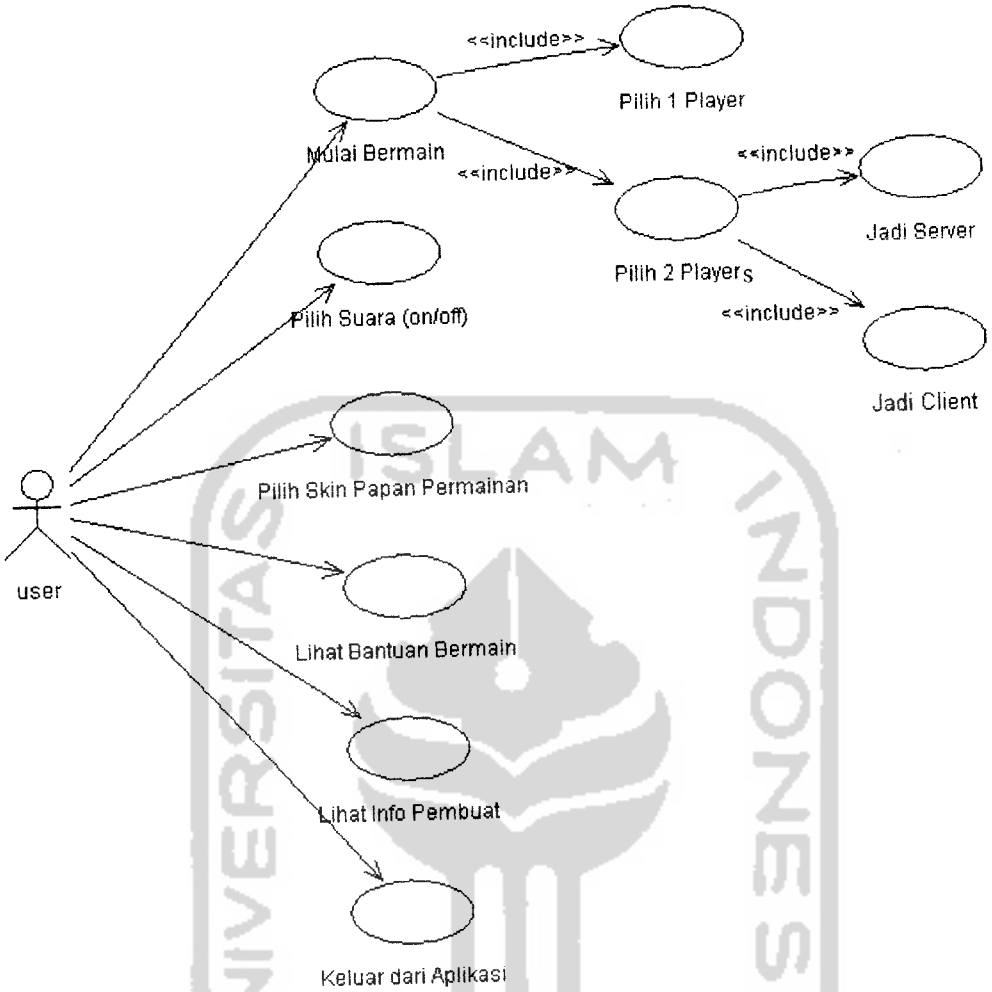
Use case diagram berisi gambaran fungsionalitas yang diharapkan dari sebuah sistem dengan fokus penekanan pada apa yang dilakukan oleh sistem, bukan bagaimana sistem melakukan sesuatu. *Use Case Diagram* menyediakan

cara untuk mendeskripsikan pandangan *eksternal* terhadap sistem dan interaksi-interaksinya dengan dunia luar. Dalam *use case diagram* ada dua pihak yang saling berhubungan, yaitu *actor* dan *use case* yang berkaitan dengan *actor*.

Pada rancangan aplikasi yang dibuat, hanya terdapat satu *actor* saja, yaitu *actor User*. Adapun *actor* tersebut berinteraksi dengan sistem melalui fungsi-fungsi yang dimiliki oleh sistem. Kemudian untuk *actor User* tersebut memiliki beberapa *use case* yang didefinisikan sebagai berikut:

a. Mulai Bermain
b. Pilih 1 *Player*
c. Pilih 2 *Players*
d. Jadi *Server*
e. Jadi *Client*
f. Pilih Suara (*on/off*)
g. Pilih *Skin* Papan Permainan
h. Lihat Bantuan Bermain
i. Lihat Info Pembuat
j. Keluar dari Aplikasi

Hubungan antar *User* dan *Use Case* lebih jelasnya dapat dilihat pada gambar 3.1.



Gambar 3.1 *Use Case Diagram*

3.2.2.2 *Class Diagram*

Class diagram digunakan untuk menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Pada *Class diagram* ini terdapat 28 *class*, yaitu:

1. *Class Object*

   Class bawaan Java ini, merupakan *class parent* dari semua *class* yang ada pada bahasa pemrograman Java, baik yang merupakan bawaan Java ataupun buatan *programmer*.

2. *Class* MIDlet

   Class MIDlet merupakan *class* utama yang harus ada pada pembuatan aplikasi mobile dengan *profile* midlet. Karena *class* inilah yang akan dibaca pertama kali oleh *mobile device*, dan juga untuk memanggil *class* lainnya. *Class* ini adalah bawaan dari Java yang merupakan *abstract class*, sehingga untuk dapat digunakan, harus diturunkan terlebih dahulu.

3. *Class* UlarTangga

   Class UlarTangga merupakan turunan dari *class* MIDlet yang melakukan semua fungsi dari *class* MIDlet tersebut. Sehingga dapat dikatakan bahwa midlet dari aplikasi permainan ular tangga ini adalah class UlarTangga.

4. *Class Canvas*

   Class ini adalah bawaan dari Java dan merupakan *class* dasar untuk membuat aplikasi yang membutuhkan penanganan *low-level event* serta untuk membuat tampilan *graphics* pada *display*.

5. *Class GameCanvas*

   Class bawaan Java ini merupakan dasar *screen* untuk menampilkan semua fungsionalitas *game* yang ada. Di dalamnya disediakan pula fasilitas untuk

melakukan sinkronisasi *graphic* dan pemanfaatan *game keys* yang berguna untuk meningkatkan performa dari sebuah *game*.

6. *Class LayoutGame*

   Class ini merupakan turunan dari *class GameCanvas*, sehingga akan menjalankan semua fungsi yang ada pada *class GameCanvas* tersebut. *Class* ini adalah *game canvas* untuk permainan satu *player* pada aplikasi permainan ular tangga.

7. *Class LayoutServer*

   Class ini juga merupakan turunan dari *class GameCanvas*, namun digunakan untuk permainan dua *player* pada aplikasi permainan ular tangga yang merupakan milik *server* (*host*).

8. *Class LayoutClient*

   Class ini juga merupakan turunan dari *class GameCanvas*. Tapi *class* ini adalah *game canvas* untuk permainan dua *player* pada aplikasi permainan ular tangga yang merupakan milik *client*.

9. *Class Select*

   Class yang diturunkan dari *class GameCanvas* ini merupakan *parent class* untuk semua *class* yang melakukan fungsi seleksi pada aplikasi permainan ular tangga.

10. *Class MenuSelect*

    Class yang merupakan turunan dari *class Select* ini digunakan untuk menampilkan menu utama pada aplikasi permainan ular tangga.

11. *Class StartGame*

    Class yang merupakan turunan dari *class Select* ini digunakan untuk menampilkan menu *Start Game* pada aplikasi permainan ular tangga.

12. *Class OptionSound*

    Class yang merupakan turunan dari *class Select* ini digunakan untuk menampilkan menu *Sound* pada aplikasi permainan ular tangga.

13. *Class OptionSkin*

    Class ini adalah turunan dari *class Select* yang digunakan untuk menampilkan menu *Skins* pada aplikasi permainan ular tangga.

14. *Class Help*

    Class ini adalah turunan dari *class Select* yang digunakan untuk menampilkan halaman *Help* pada aplikasi permainan ular tangga.

15. *Class About*

    Class ini adalah turunan dari *class Select* yang digunakan untuk menampilkan halaman *About* pada aplikasi permainan ular tangga.

16. *Class Exit*

    Class Exit digunakan untuk menampilkan halaman konfirmasi sebelum keluar dari aplikasi permainan ular tangga. *Class* ini adalah turunan dari *class Select*.

17. *Class ClientServer*

    Class ini adalah turunan dari *class Select* yang digunakan untuk menampilkan menu 2 *Players* pada aplikasi permainan ular tangga.

18. *Class EndContinue*

    Class EndContinue digunakan untuk menampilkan halaman konfirmasi saat aplikasi permainan ular tangga sedang dalam keadaan *pause*. *Class* ini adalah turunan dari *class Select*.

19. *Class VolumeSound*

    Class yang merupakan turunan dari *class Select* ini digunakan untuk menampilkan halaman *Set Volume* pada aplikasi permainan ular tangga.

20. *Class Splash*

    Class ini merupakan turunan dari *class Canvas* yang digunakan untuk menampilkan *splash screen* pada aplikasi permainan ular tangga.

21. *Class Layer*

    Class bawaan Java ini merupakan representasi dari visual elemen yang ditampilkan dalam sebuah *game*. *Class* ini merupakan dasar bagi pengaturan lapisan gambar seperti pengaturan posisi, ukuran dan tampilan.

22. *Class Sprite*

    Class bawaan Java ini adalah dasar untuk menghasilkan sebuah animasi pada *layer*. *Class* ini memungkinkan pembuatan sebuah transformasi dan deteksi tubrukan sebuah gambar atau *layer* dengan gambar atau *layer* lain.

23. *Class* Bidak

    Class yang merupakan turunan dari *class Sprite* ini adalah representasi dari karakter bidak.

24. *Class* Dadu

    Class yang merupakan turunan dari *class Sprite* ini adalah representasi dari karakter dadu.

25. *Class* KocokanDadu

    Class ini adalah turunan dari *class Sprite* yang digunakan untuk merepresentasikan karakter kocokan dadu.

26. *Class Bluetooth*

    Class yang menggunakan *package* javax.*bluetooth* ini digunakan untuk menginisialisasikan atribut dan metode yang diperlukan dalam melakukan koneksi menggunakan *bluetooth.*

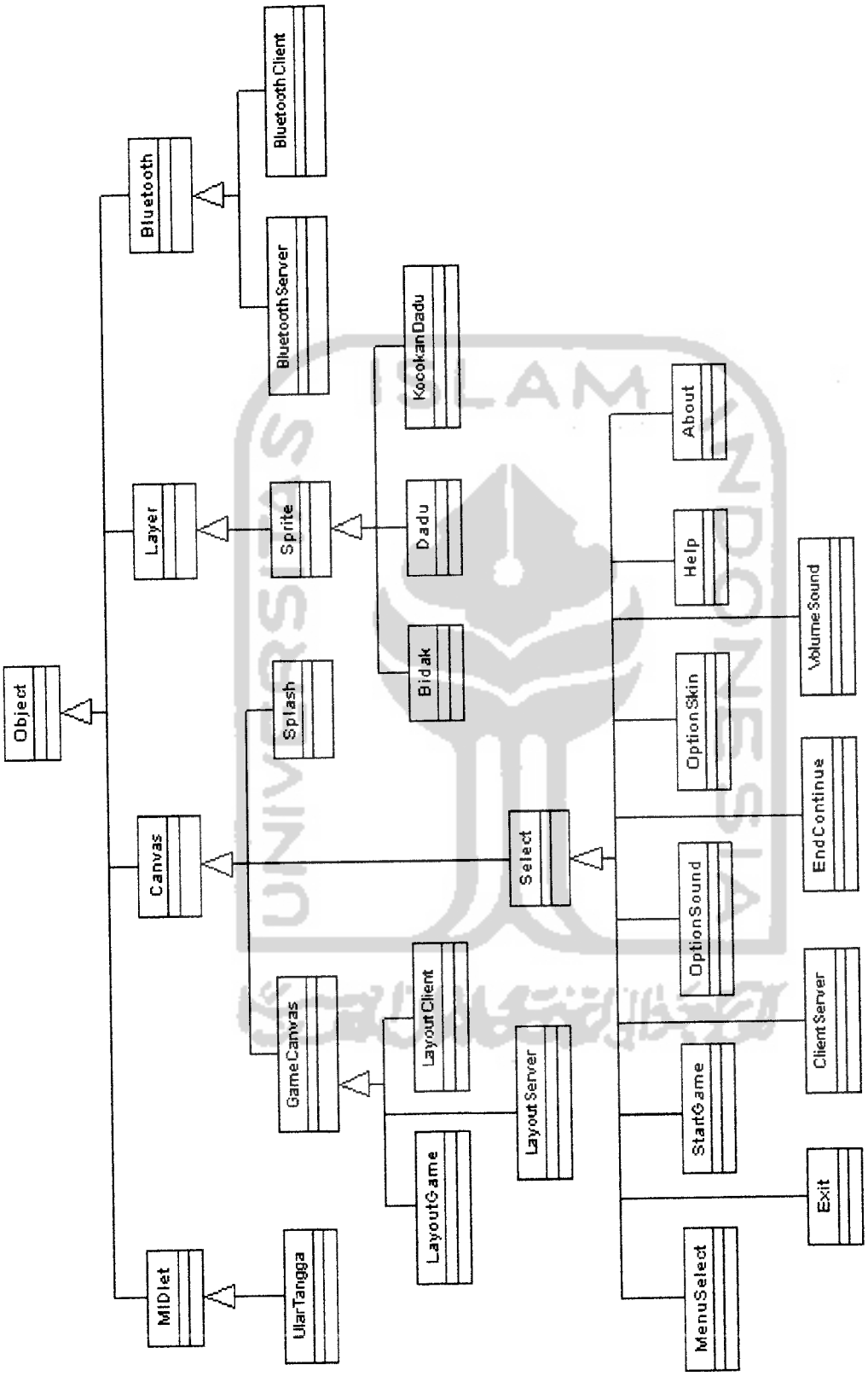
27. *Class BluetoothServer*

    Class ini merupakan turunan dari *class Bluetooth* yang akan digunakan sebagai *server.*

28. *Class BluetoothClient*

    Class yang merupakan turunan dari class *Bluetooth* ini digunakan sebagai *client.*

Class diagram dari aplikasi permainan ular tangga ini dapat dilihat pada gambar 3.2



Gambar 3.2 *Class Diagram*

3.2.2.3 *Sequence Diagram*

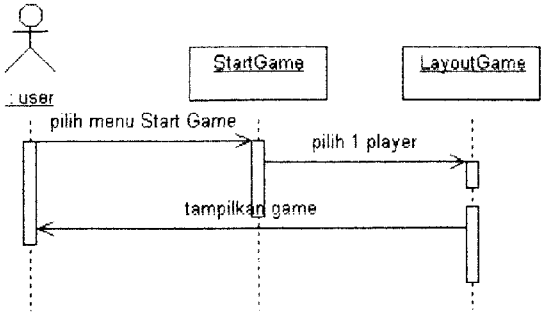
Sequence Diagram menggambarkan perilaku sistem secara dinamis dan memperlihatkan interaksi dari obyek-obyek. Interaksi antar obyek dapat disusun berdasarkan urutan waktu yang menunjukkan skenario dan urut-urutan pertukaran data.

Dari tahapan analisis kebutuhan yang dilakukan sebelumnya maka dapat dibentuk beberapa *sequence diagram* untuk menunjukkan urutan-urutan proses dari masing-masing *use case*. Pada aplikasi permainan Ular Tangga ini terdapat beberapa *Sequence Diagram*, yaitu:

a. *Sequence Diagram* untuk mulai bermain satu *player*
b. *Sequence Diagram* untuk mulai bermain dua *player* dan menjadi *server*
c. *Sequence Diagram* untuk mulai bermain dua *player* dan menjadi *client*
d. *Sequence Diagram* untuk pengaturan suara
e. *Sequence Diagram* untuk memilih *skin* papan permainan
f. *Sequence Diagram* untuk melihat bantuan permainan
g. *Sequence Diagram* untuk melihat informasi pembuat

3.2.2.3.1 ***Sequence Diagram*** untuk skenario **Mulai Bermain Satu *Player***

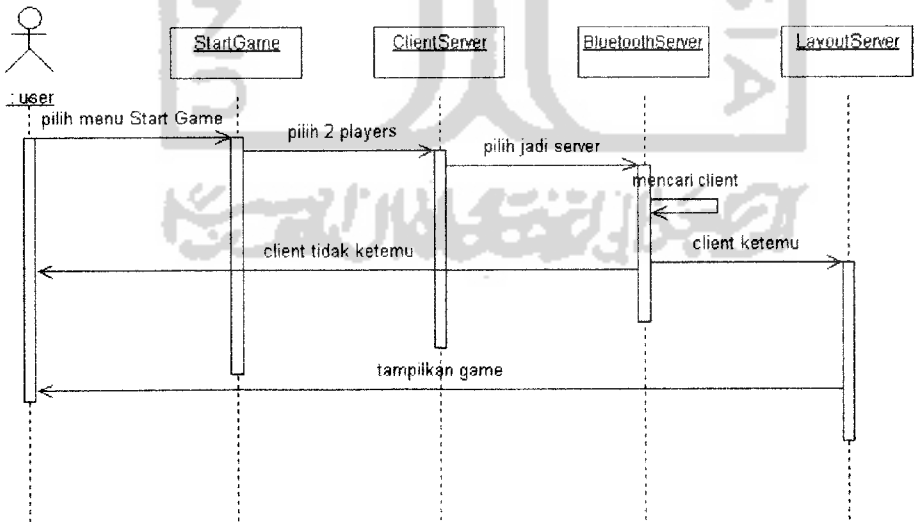
Sequence diagram ini menunjukkan proses pada saat *user* mulai bermain satu *player*. Gambar 3.3 menunjukkan *sequence diagram* untuk skenario mulai bermain satu *player* yang dilakukan oleh *user*.



Gambar 3.3 *sequence diagram* untuk skenario mulai bermain satu *player*

3.2.2.3.2 *Sequence Diagram* untuk skenario **Mulai Bermain Dua *Player* dan Menjadi *Server***

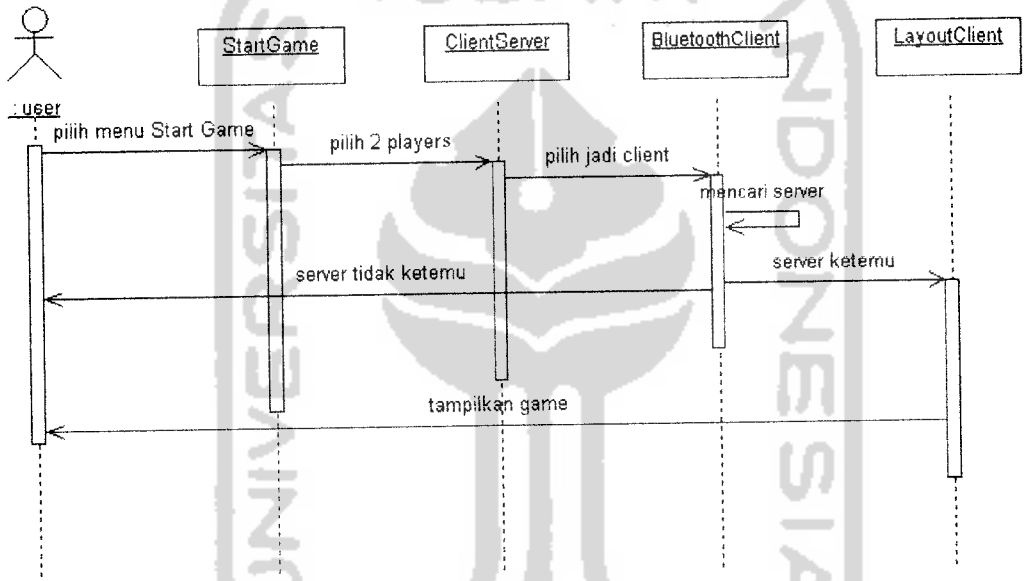
Sequence diagram ini menunjukkan proses pada saat *user* mulai bermain dua *player* dan kemudian memilih untuk menjadi *server*. Gambar 3.4 menunjukkan *sequence diagram* untuk skenario mulai bermain dua *player* dan menjadi *server* yang dilakukan oleh *user*.



Gambar 3.4 *sequence diagram* untuk skenario mulai bermain dua *player* dan menjadi *server*

3.2.2.3.3 *Sequence Diagram* untuk skenario **Mulai Bermain Dua *Player* dan Menjadi *Client***

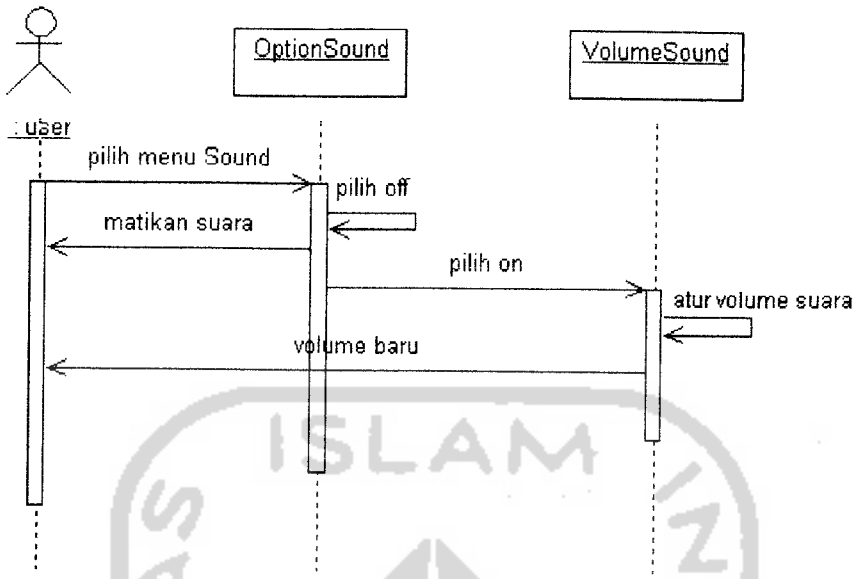
Sequence diagram ini menunjukkan proses pada saat *user* mulai bermain dua *player* dan kemudian memilih untuk menjadi *client*. Gambar 3.5 menunjukkan *sequence diagram* untuk skenario mulai bermain dua *player* dan menjadi *client* yang dilakukan oleh *user*.



Gambar 3.5 *sequence diagram* untuk skenario mulai bermain dua *player* dan menjadi *client*

3.2.2.3.4 *Sequence Diagram* untuk skenario **Pengaturan Suara**

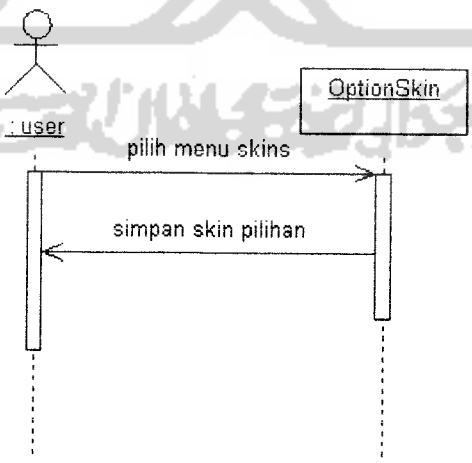
Sequence diagram ini menunjukkan proses pada saat *user* melakukan pengaturan suara. Gambar 3.6 menunjukkan *sequence diagram* untuk skenario pengaturan suara yang dilakukan oleh *user*.



Gambar 3.6 *sequence diagram* untuk skenario pengaturan suara

3.2.2.3.5 *Sequence Diagram* untuk skenario **Memilih *Skin* Papan Permainan**

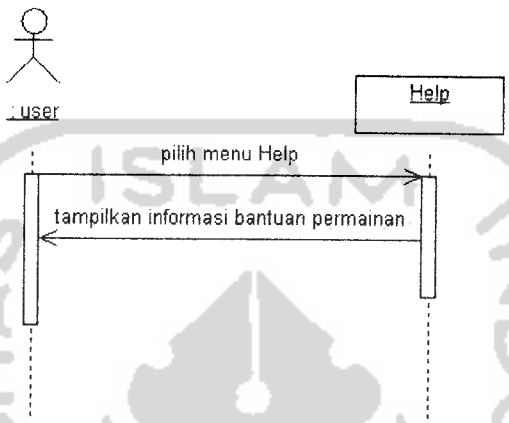
Sequence diagram ini menunjukkan proses pada saat *user* memilih *skin* papan permainan yang diinginkan. Gambar 3.7 menunjukkan *sequence diagram* untuk skenario memilih *skin* papan permainan yang dilakukan oleh *user*.



Gambar 3.7 *sequence diagram* untuk skenario memilih *skin* papan permainan

3.2.2.3.6 *Sequence Diagram* untuk skenario **Melihat Bantuan Permainan**

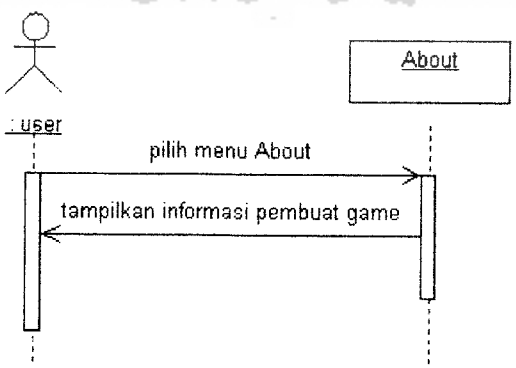
Sequence diagram ini menunjukkan proses pada saat *user* memilih untuk melihat bantuan permainan. Gambar 3.8 menunjukkan *sequence diagram* untuk skenario melihat bantuan permainan yang dilakukan oleh *user*.



Gambar 3.8 *sequence diagram* untuk skenario melihat bantuan permainan

3.2.2.3.7 *Sequence Diagram* untuk skenario **Melihat Informasi Pembuat**

Sequence diagram ini menunjukkan proses pada saat *user* memilih untuk melihat informasi pembuat aplikasi. Gambar 3.9 menunjukkan *sequence diagram* untuk skenario melihat informasi pembuat yang dilakukan oleh *user*.



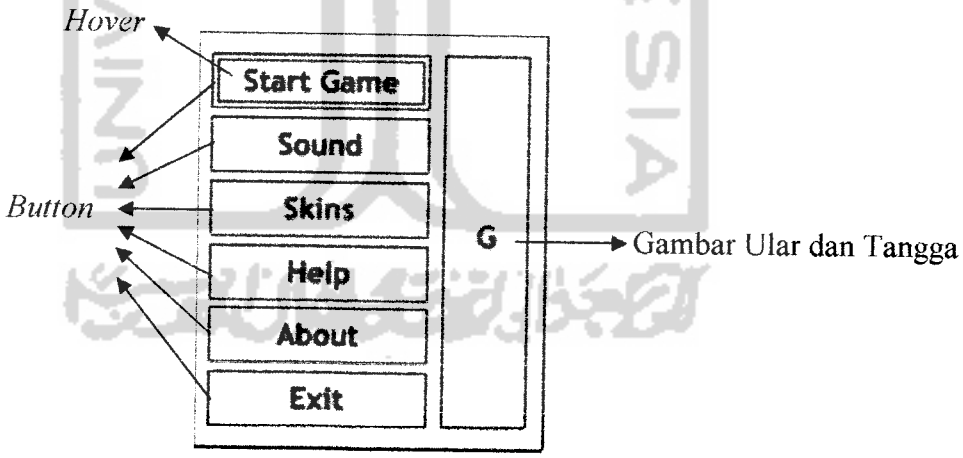
Gambar 3.9 *sequence diagram* untuk skenario melihat informasi pembuat

3.2.3 Perancangan Antar Muka (*Interface*)

Perancangan *Interface* menggambarkan desain tampilan dari sistem, ilustrasi dari rancangan *interface* terhadap sistem yang akan diaplikasikan. Desain *interface* yang utama ditujukan kepada *user*, dimana *interface* didesain sedemikian rupa untuk memudahkan penggunaan sistem aplikasi ini.

3.2.3.1 Rancangan Antar Muka Menu Utama

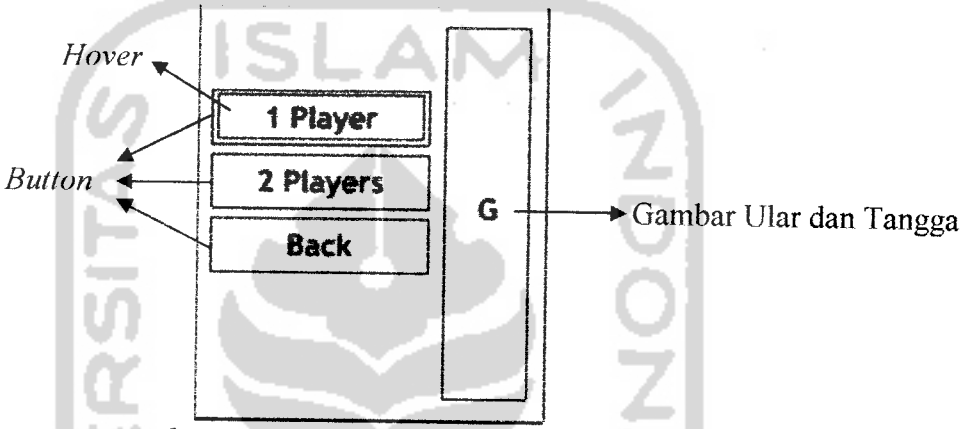
Antar muka ini merupakan menu pertama yang ditampilkan ketika *user* pertama kali masuk ke dalam aplikasi setelah *splash screen*. Di menu utama ini, *user* dapat memilih untuk mulai bermain, mengatur suara, memilih *skin* papan permainan, melihat bantuan permaianan, melihat info pembuat atau keluar dari aplikasi. Rancangan antar muka menu utama dapat dilihat pada gambar 3.10



Gambar 3.10 Menu Utama

3.2.3.2 Rancangan Antar Muka Menu *Start Game*

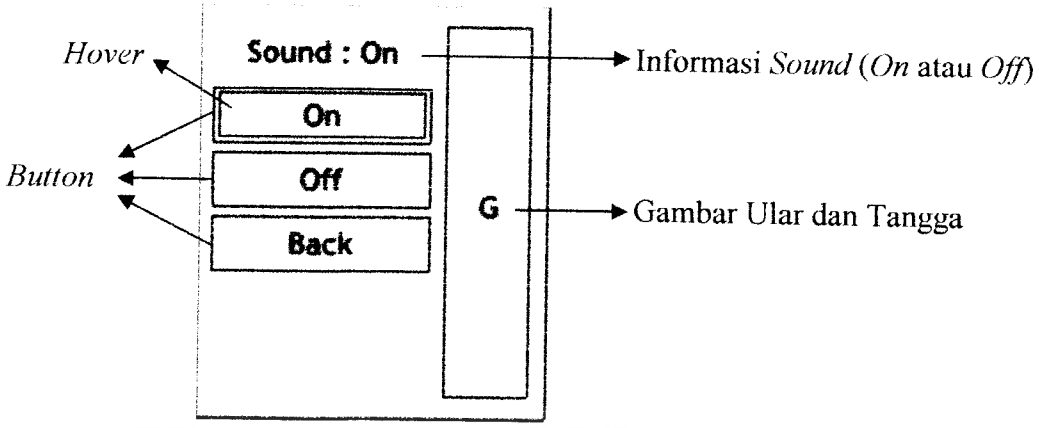
Antar muka ini merupakan menu yang ditampilkan ketika *user* memilih menu *Start Game* pada menu utama. Pada halaman menu ini, *user* dapat memilih untuk memulai permainan dengan satu pemain (lawan CPU), dua pemain atau kembali ke menu utama. Rancangan antar muka menu *Start Game* dapat dilihat pada gambar 3.11



Gambar 3.11 Menu *Start Game*

3.2.3.3 Rancangan Antar Muka Menu *Sound*

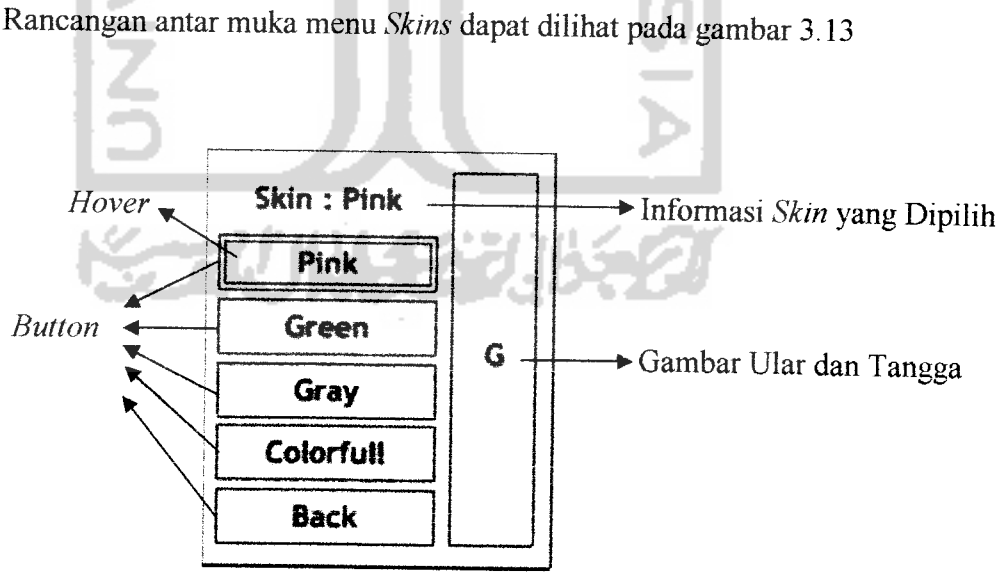
Antar muka ini merupakan menu yang ditampilkan ketika *user* memilih menu *Sound* pada menu utama. Pada halaman menu ini, *user* dapat memilih untuk menghidupkan suara (*on*), mematikan suara (*off*) atau kembali ke menu utama. Rancangan antar muka menu *Sound* dapat dilihat pada gambar 3.12



Gambar 3.12 Menu *Sound*

3.2.3.4 Rancangan Antar Muka Menu *Skins*

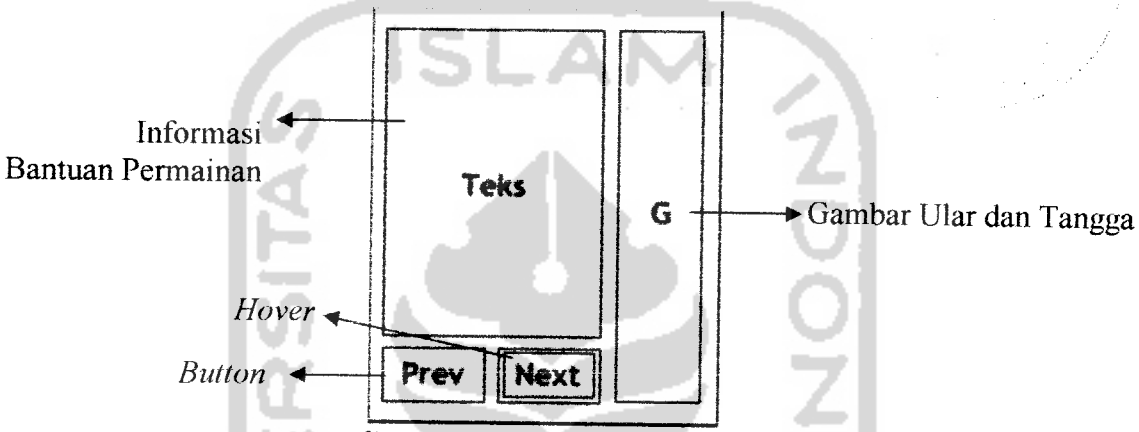
Antar muka ini merupakan menu yang ditampilkan ketika *user* memilih menu *Skins* pada menu utama. Pada halaman menu ini, *user* dapat memilih warna *skin* yang diinginkan atau kembali ke menu utama. Warna skin yang dapat dipilih yaitu merah muda (*pink*), hijau (*green*), abu-abu (*gray*), dan warna-warni (*colorfull*). Rancangan antar muka menu *Skins* dapat dilihat pada gambar 3.13



Gambar 3.13 Menu *Skins*

3.2.3.5 Rancangan Antar Muka Halaman *Help*

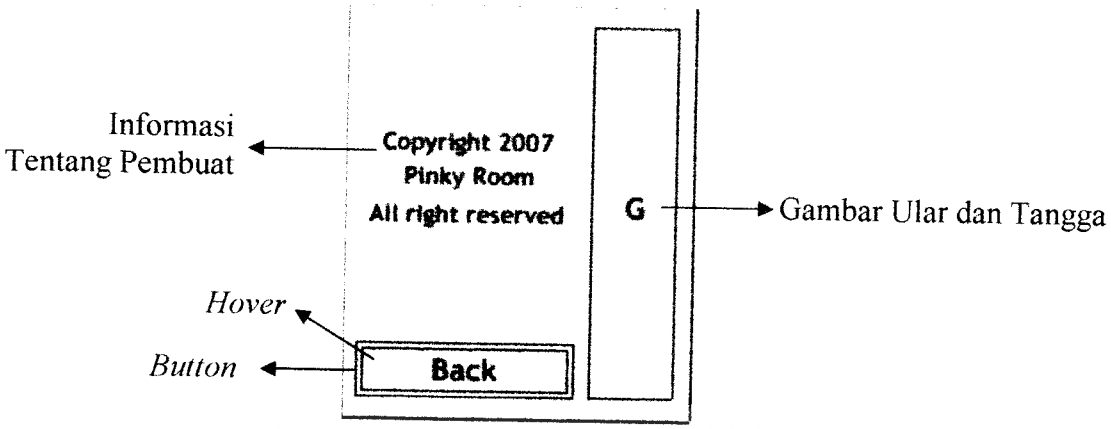
Antar muka ini merupakan halaman yang ditampilkan ketika *user* memilih menu *Help* pada menu utama. Pada halaman ini, *user* dapat dapat melihat bantuan permainan atau kembali ke menu utama. Rancangan antar muka menu *Help* dapat dilihat pada gambar 3.14



Gambar 3.14 Halaman *Help*

3.2.3.6 Rancangan Antar Muka Halaman *About*

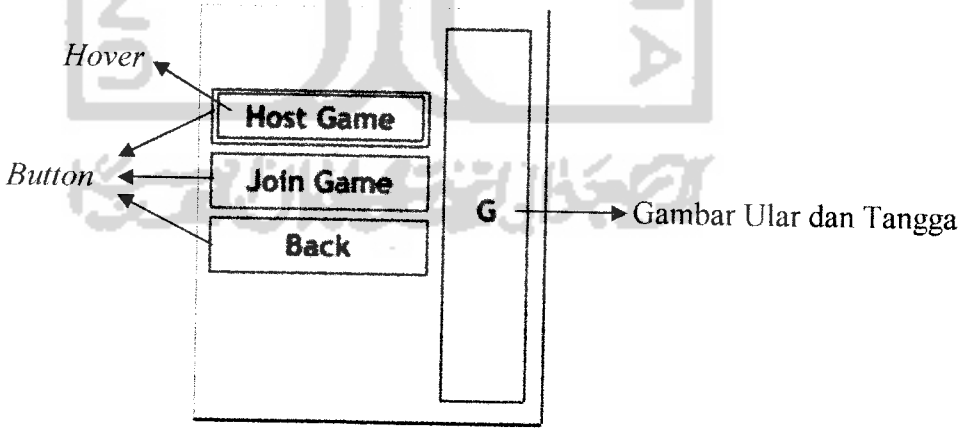
Antar muka ini merupakan halaman yang ditampilkan ketika *user* memilih menu *About* pada menu utama. Pada halaman ini, *user* dapat dapat melihat informasi tentang pembuat aplikasi atau kembali ke menu utama. Rancangan antar muka menu *About* dapat dilihat pada gambar 3.15



Gambar 3.15 Halaman *About*

3.2.3.7 Rancangan Antar Muka Menu *2 Players*

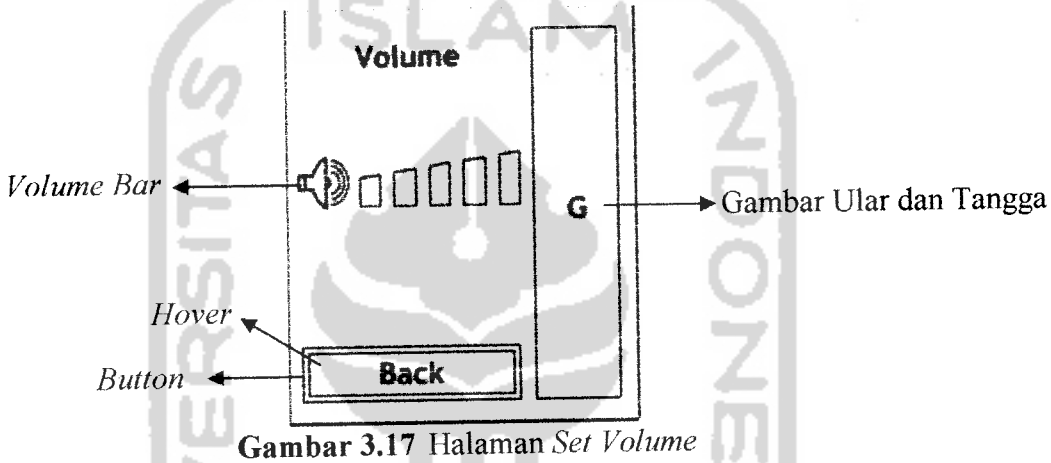
Antar muka ini merupakan menu yang ditampilkan ketika *user* memilih menu *2 Players* pada menu *Start Game*. Pada halaman menu ini, *user* dapat memilih untuk menjadi *server* (*host game*), menjadi *client* (*join game*) atau kembali ke menu *Start Game*. Rancangan antar muka menu *2 Players* dapat dilihat pada gambar 3.16



Gambar 3.16 Menu *2 Players*

3.2.3.8 Rancangan Antar Muka Halaman *Set Volume*

Antar muka ini merupakan halaman yang ditampilkan ketika *user* memilih menu *On* pada menu *Sound*. Pada halaman ini, *user* dapat dapat mengatur *volume* suara atau kembali ke menu *Sound*. Rancangan antar muka menu *Set Volume* dapat dilihat pada gambar 3.17

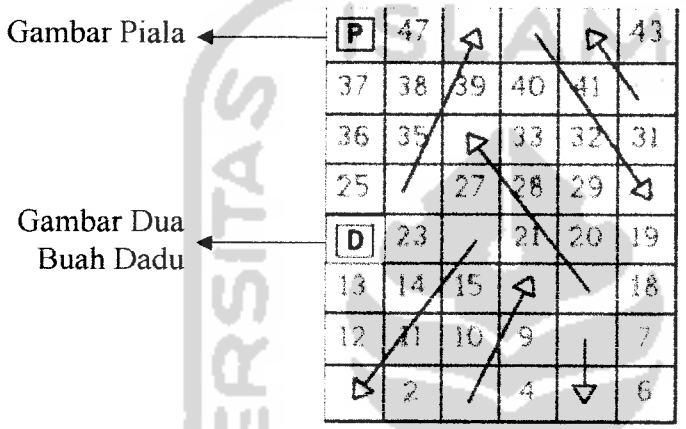


Gambar 3.17 Halaman *Set Volume*

3.2.3.9 Rancangan Antar Muka Papan Permainan

Antar muka ini merupakan halaman yang ditampilkan ketika *user* telah memilih untuk mulai bermain (satu pemain atau pun dua pemain). Pada papan permainan ini terdapat 48 kotak, enam kotak setiap barisnya, dan delapan kotak untuk setiap kolomnya. Terdapat empat buah tangga yaitu dari kotak "3" ke kotak "16", dari kotak "17" ke kotak "34", dari kotak "26" ke kotak "46", dan dari kotak "42" ke kotak "44". Sedangkan ular ada tiga buah yaitu dari kotak "8" ke kotak "5", dari kotak "22" ke kotak "1", dan dari kotak "45" ke kotak "30". Pada gambar 3.17, tangga diberi simbol panah ke atas sedangkan ular diberi simbol panah ke bawah.

Pada papan permainan ini terdapat gambar piala (pada kotak 48) dan dua buah dadu (pada kotak 24). Gambar piala menunjukkan kotak terakhir dimana bidak (pion) yang berhasil mencapainya terlebih dahulu adalah pemenangnya. Sedangkan gambar dua buah dadu menunjukkan bahwa bidak (pion) yang berhenti pada kotak ini berhak untuk mengocok dadu kembali.



Gambar 3.18 Papan Permainan

3.3 Implementasi Perangkat Lunak

Implementasi merupakan tahap dimana sistem siap dioperasikan pada tahap yang sebenarnya, sehingga akan diketahui apakah sistem yang telah dibuat benar-benar sesuai dengan yang direncanakan. Pada implementasi perangkat lunak ini akan dijelaskan bagaimana program sistem ini bekerja, dengan memberikan tampilan yang telah dibuat.

3.3.1 Batasan Implementasi

Implementasi merupakan tahap dimana sistem siap diaplikasikan dalam keadaan sesungguhnya. Dari implementasi akan diketahui apakah sistem yang dibuat benar-benar dapat berjalan dan menghasilkan keluaran yang sesuai dengan perancangan yang disiapkan. Sebelum program diterapkan dan diimplementasikan, maka program harus bebas kesalahan (*error free*). Kesalahan program yang mungkin terjadi antara lain kesalahan penulisan bahasa, kesalahan sewaktu proses atau kesalahan logikal.

Implementasi yang dibuat adalah sebuah *game* untuk *mobile device*, yaitu *game* Ular Tangga yang dapat dimainkan oleh satu pemain saja (lawan CPU) atau pun oleh dua orang pemain menggunakan koneksi *bluetooth*. Untuk dapat memainkan permainan dengan dua orang pemain, kedua *mobile device* selain memiliki konektifitas *bluetooth*, juga harus dilengkapi dengan Java APIs *for Bluetoth Wireless Technology* (JABWT / JSR-82) dan CLDC 1.1.

3.3.2 Implementasi Antar Muka

Implementasi dari aplikasi ini terdiri dari beberapa menu yang memiliki fungsinya masing-masing. Menu-menu tersebut akan tampil secara berurutan sesuai dengan urutan yang telah dibuat, setelah *user* melakukan proses tertentu.

3.3.2.1 Halaman *Splash Screen*

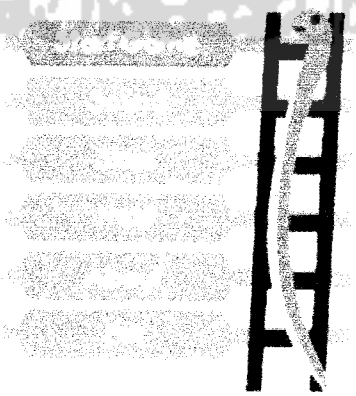
Halaman *Splash Screen* merupakan halaman yang pertama kali ditampilkan ketika *user* masuk ke dalam aplikasi. Halaman ini, akan berganti

dengan menu Utama secara otomatis setelah satu setengah detik. Tampilan halaman *Splash Screen* dapat dilihat pada gambar 3.19

Gambar 3.19 Tampilan *Splash Screen*

3.3.2.2 Menu Utama

Menu Utama merupakan menu pertama yang ditampilkan setelah *Splash Screen*. Di menu Utama ini, *user* dapat memilih untuk mulai bermain, mengatur suara, memilih *skin* papan permainan, melihat bantuan permaianan, melihat info pembuat atau keluar dari aplikasi. Tampilan menu Utama dapat dilihat pada gambar 3.20



Gambar 3.20 Tampilan Menu Utama

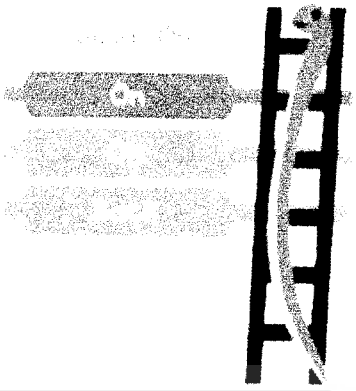
3.3.2.3 Menu *Start Game*

Menu *Start Game* ditampilkan setelah *user* memilih "*Start Game*" pada menu Utama. Terdapat tiga pilihan dalam menu ini, yaitu bermain satu *player* (1 Player), dua *player* (2 *Players*) atau kembali lagi ke menu Utama (*Back*). Tampilan menu *Start Game* dapat dilihat pada gambar 3.21

Gambar 3.21 Tampilan Menu *Start Game*

3.3.2.4 Menu *Sound*

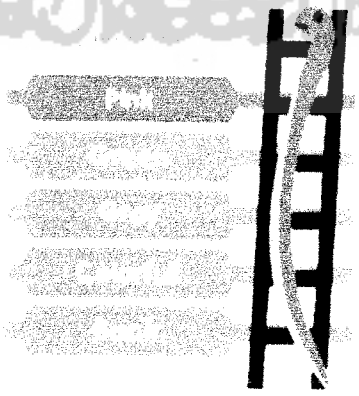
Menu *Sound* ditampilkan setelah *user* memilih "*Sound*" pada menu Utama. Terdapat tiga pilihan dalam menu ini, yaitu menghidupkan suara, mematikan suara atau kembali lagi ke menu Utama. Bila *user* memilih untuk menghidupkan suara, maka halaman *Set Volume* akan ditampilkan. Tampilan menu *Sound* dapat dilihat pada gambar 3.22



Gambar 3.22 Tampilan Menu *Sound*

3.3.2.5 Menu *Skins*

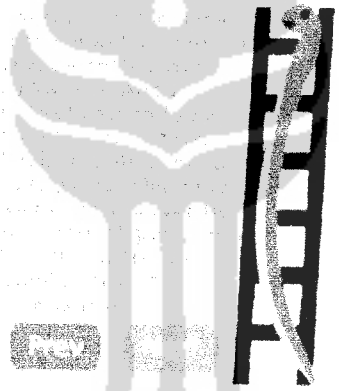
Apabila *user* memilih "*Skins*" pada menu Utama, maka menu *Skins* akan ditampilkan. Di menu ini, *user* dapat mengganti warna *skin* papan permainan ular tangga, dengan warna-warna yeng telah disediakan, yaitu: merah muda, hijau, abu-abu, dan warna-warni. Bila *user* tidak memilih dalam menu ini, maka warna *skin* yang akan ditampilkan adalah merah muda. Tampilan menu *Skins* dapat dilihat pada gambar 3.23



Gambar 3.23 Tampilan Menu *Skins*

3.3.2.6 Halaman *Help*

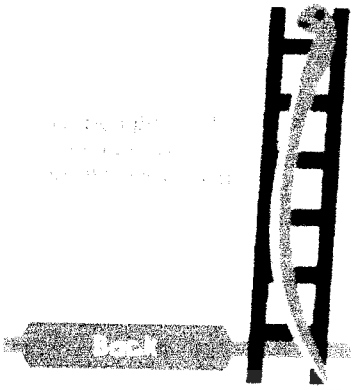
Halaman *Help* berisi tentang bantuan permainan. Halaman ini terdiri dari dua halaman. Pada halaman pertama bila *user* memilih "*Next*", maka akan masuk ke halaman dua, dan bila *user* memilih "*Prev*" maka akan kembali ke menu Utama. Sedangkan pada halaman dua, bila *user* memilih "*Next*", maka akan kembali ke menu Utama, dan bila *user* memilih "*Prev*" maka akan kembali ke halaman pertama. Halaman *Help* ini akan ditampilkan ketika *user* memilih "*Help*" pada menu Utama. Tampilan halaman *Help* dapat dilihat pada gambar 3.24



Gambar 3.24 Tampilan Halaman *Help*

3.3.2.7 Halaman *About*

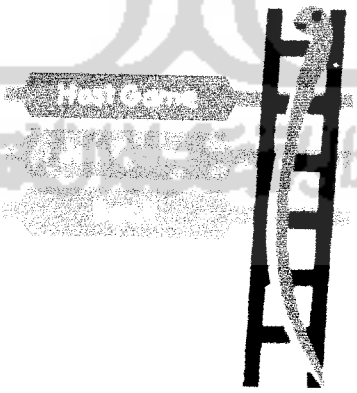
Halaman *About* ditampilkan setelah item "About" pada menu Utama dipilih. Halaman ini menampilkan informasi tentang pembuat aplikasi permainan ular tangga ini. Tampilan halaman *About* dapat dilihat pada gambar 3.25



Gambar 3.25 Tampilan Halaman *About*

3.3.2.8 Menu 2 *Players*

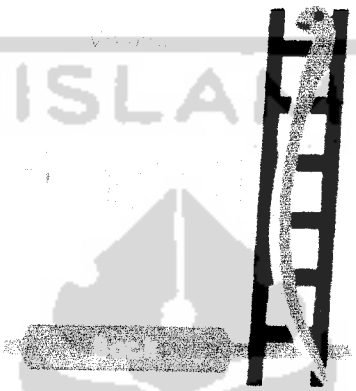
Apabila *user* memilih "2 *Players*" pada menu *Start Game*, maka menu ini akan ditampilkan. Di menu ini, *user* dapat memilih untuk menjadi *server* (*Host Game*) atau *client* (*Join Game*). Untuk kembali lagi ke menu *Start Game user* dapat memilih item "*Back*". Tampilan menu 2 *Players* dapat dilihat pada gambar 3.26



Gambar 3.26 Tampilan Menu 2 *Players*

3.3.2.9 Halaman *Set Volume*

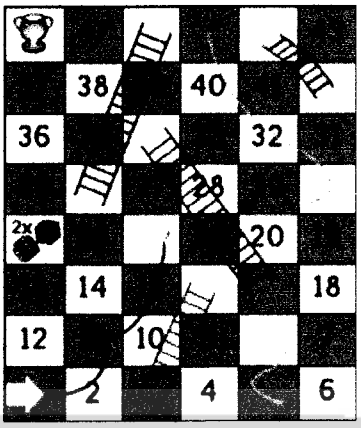
Pada halaman *Set Volume*, *user* dapat mengatur *volume* suara. Untuk membesarkan suara *user* cukup menekan *key right*, sedangkan untuk mengecilkan *user* cukup menekan *key left*. Tampilan halaman *Set Volume* dapat dilihat pada gambar 3.27



Gambar 3.27 Tampilan Halaman *Set Volume*

3.3.2.10 Papan Permainan

Papan permainan ditampilkan bila *user* memilih untuk mulai bermain, baik hanya satu pemain atau pun dua pemain. Warna *skin* papan permainan tergantung dari *skin* yang dipilih oleh *user* pada menu *Skins*. Tampilan papan permainan dengan warna *skin* merah muda dapat dilihat pada gambar 3.28



Gambar 3.28 Tampilan Papan Permainan

3.3.3 Implementasi Prosedural

Implementasi prosedural merupakan penerapan rancangan yang telah dibuat kedalam bentuk program (*sourcecode*). Pada implementasi prosedural ini hanya akan dijelaskan tentang fungsi-fungsi apa saja yang digunakan, yaitu:

1. Fungsi untuk mengocok dadu.
2. Fungsi untuk menghentikan kocokan dadu.
3. Fungsi untuk menjalankan bidak (pion).
4. Fungsi untuk mengecek apakah bidak (pion) bertemu dengan ular atau tangga.
5. Fungsi untuk menjalankan bidak (pion) mengikuti tangga atau ular.
6. Fungsi jalan mundur pada bidak (pion) yang telah berada pada kotak terakhir, sementara nilai dadu berlebih.
7. Fungsi untuk mengecek apakah bidak (pion) telah memenangkan permainan.

3.3.3.1 Fungsi untuk Mengocok Dadu

Fungsi untuk mengocok dadu ini terdapat di dalam *class* Dadu. Berikut adalah *method* untuk mengocok dadu yaitu:

```
public void gerakan(){
   setFrame(currentFrame);
   if (currentFrame == 1)
      currentFrame=0;
   else
      currentFrame++;
}
```

Namun sebelumnya *variable* `currentFrame` diberi nilai "0" pada saat inisialisasi.

3.3.3.2 Fungsi untuk Menghentikan Kocokan Dadu

Fungsi untuk menghentikan kocokan dadu ini terdapat di dalam *class* *LayoutGame*. Berikut adalah *method* untuk menghentikan kocokan dadu:

```
private void daduStop()
{
   int keyStates = getKeyStates();
   if ((keyStates & FIRE_PRESSED) != 0)
   {
      n = i+1;
      tekan = true;
      alert1.setVisible(false);
   }
}
```

3.3.3.3 Fungsi untuk Menjalankan Bidak (Pion)

Fungsi untuk menjalankan bidak (pion) ini terdapat di dalam *class* Bidak. Berikut adalah *method* untuk menjalankan bidak (pion):

```
public void turn ()
{
   pos_x = getX();
   pos_y = getY();
   int n = 1;
   if (langkah == 48)
   {
      langkah = 46;
```

```
        pos_x+= 30;
        mundur = true;
    }else if (((pos_x == -27)&&(pos_y == 185)) ||
        ((pos_x < 148)&&(gerakan.equals("kanan") )))
    {
        pos_x+=30;
        pos_y = getY();
        gerakan="kanan";
    }else if (langkah%6 == 0)
    {
        pos_y+=-26;
        pos_x+=0;
        if (pos_x > 148)
            gerakan="kiri";
        else if (pos_x < 28)
            gerakan="kanan";
    }else if ((pos_x > 28)&&(gerakan == "kiri"))
    {
        pos_x+=-30;
        pos_y+= 0;
        gerakan="kiri";
    }
    langkah++;
    setPosition(pos_x, pos_y);
    hold(500);
}
```

3.3.3.4 Fungsi untuk Mengecek Apakah Bidak (Pion) Bertemu Ular atau Tangga

Fungsi ini terdapat di dalam *class* Bidak. Berikut adalah *method* untuk mengecek apakah bidak (pion) bertemu dengan ular atau tangga:

```
public void cekUlarTangga(int lkh)
{
    int j = 0;
    int k = 1;
    int l = 2;
    int m = 3;
    for (int i=0;i<7;i++)
    {
        if (lkh == ut[i][j])
        {
            tam_x = 30*ut[i][k];
            tam_y = 26*ut[i][l];
            langkah2 = ut[i][m];
            cek = true;
        }
    }
}
```

3.3.3.5 Fungsi untuk Menjalankan Bidak (Pion) Mengikuti Tangga atau Ular

Fungsi ini terdapat di dalam *class* Bidak. Fungsi ini dijalankan apabila bidak bertemu ular atau tangga. Berikut adalah *method* dari fungsi ini:

```
public void turn2()
{
   pos_x+=tam_x/5;
   pos_y+=tam_y/5;
   setPosition(pos_x, pos_y);
   hold(500);

   langkah = langkah2;

   if (((langkah-1)/6) % 2 == 0)
   {
      gerakan = "kanan";
   }else{
      gerakan = "kiri";
   }
   cek = false;
}
```

3.3.3.6 Fungsi Jalan Mundur pada Bidak (Pion)

Fungsi ini dijalankan apabila bidak telah sampai di kotak terakhir, tetapi nilai dadunya masih tersisa (berlebih). Fungsi ini terdapat di dalam *class* Bidak. Berikut adalah *method* dari fungsi jalan mundur:

```
public void mundur()
{
   if ((langkah-1)%6 == 0)
   {
      pos_y+=26;
      if(((langkah-1)/6)%2 == 1)
         gerakan="kiri";
      else
         gerakan="kanan";
   }
   else if (((langkah-1)/6)%2 == 1)
   {
      pos_x+=30;
      gerakan="kiri";
   }
   else
   {
      pos_x+=-30;
      gerakan="kanan";
```

```
    }
    langkah--;
    setPosition(pos_x, pos_y);
    hold(500);
}
```

3.3.3.7 Fungsi untuk Mengecek Apakah Bidak (Pion) Telah Memenangkan Permainan

Fungsi ini digunakan untuk mengecek apakah bidak berhenti pada kotak terakhir pada saat nilai dadu habis (tidak berlebih). Fungsi ini terdapat di dalam *class LayoutGame*. Kutipan *source code* dari fungsi ini yaitu:

```
switch(giliran)
{
   case 1: if (i<=7){
              alert3.setVisible(true);
              i++;
           }else{
              i = 1;
              giliran = 3;
           }
           break;
   case 2: if (i<=7){
              alert4.setVisible(true);
              i++;
           }else{
              i = 1;
              giliran = 3;
           }
           break;
   case 3: stop();
}
```

BAB IV

HASIL DAN PEMBAHASAN

4.1 Pengujian Program

Pada tahap analisis kinerja perangkat lunak akan dilakukan pengujian aplikasi untuk menganalisis kinerja aplikasi permainan ular tangga menggunakan koneksi *bluetooth* dengan J2ME. Dari pengujian akan diketahui apakah fungsi-fungsi yang ada dalam aplikasi ini dapat berjalan dengan baik dan memenuhi kebutuhan. Pengujian kinerja aplikasi ini dilakukan untuk mengetahui kesalahan-kesalahan yang ada dan juga untuk mengetahui upaya penanganan kesalahannya.

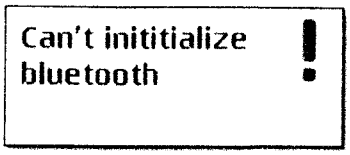
Beberapa pengujian memiliki dua kemungkinan, yaitu prosedur normal (benar) dan prosedur tidak normal (salah). Pengujian pada prosedur tidak normal digunakan untuk melihat apakah aplikasi mampu menangani kesalahan dengan baik serta mampu mengkomunikasikannya kepada pemakai aplikasi untuk kemudian memberikan saran atau solusi kepada pemakai aplikasi.

4.2 Analisis Kinerja Sistem

4.2.1 Penanganan Kesalahan

Apabila terjadi kesalahan pada saat menjalankan aplikasi, maka sistem akan memberikan tanggapan (*feedback*) kepada *user*. Adapun penanganan kesalahan yang terdapat dalam aplikasi ini adalah penanganan kesalahan bila *user* belum menyalakan *bluetooth*. Apabila user belum menyalakan *bluetooth* pada saat

memilih untuk bermain dua *player*, maka akan muncul pesan *error* seperti pada gambar 4.1



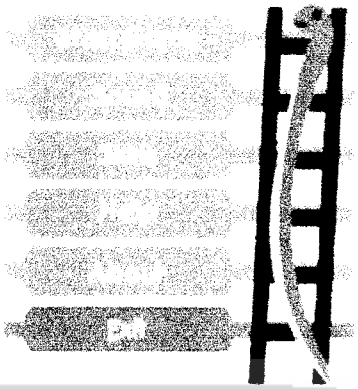
Gambar 4.1 Tampilan pesan *error* bila *bluetooth* belum dinyalakan

4.2.2 Pengujian dan Analisis

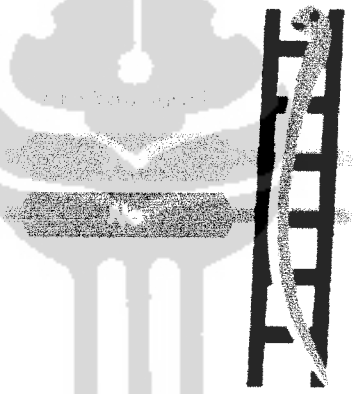
Pada tahap pengujian dan analisis program ini, dilakukan pembandingan antara kebenaran masukan serta kesesuaian program dengan kebutuhan sistem. Di dalam sistem ini, masukan hanya berupa *keypad* yang ditekan oleh *user*. *Keypad up* untuk menggerakkan ke atas, *down* untuk menggerakkan ke bawah, *right* untuk menggerakkan ke kanan, *left* untuk menggerakkan ke kiri, dan *center* untuk memilih (*select* / *enter*).

1. Pada Menu Utama

   Pada menu utama ini masukan yang dapat diterima adalah pada saat *user* menekan *keypad up*, *down* atau *center*. Contoh masukan yang dimasukkan untuk menguji keluaran output yang dihasilkan yaitu menekan *keypad down* sebanyak lima kali kemudian menekan *keypad center*. Hasil masukan pada menu utama tersebut dapat dilihat pada gambar 4.2 dan 4.3



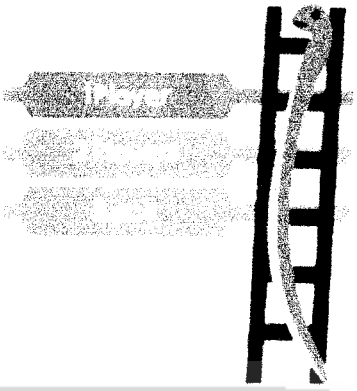
Gambar 4.2 Antar muka menu Utama ketika *keypad down* ditekan lima kali



Gambar 4.3 Antar muka menu Utama ketika *keypad center* kemudian ditekan

2. Pada Menu *Start Game*

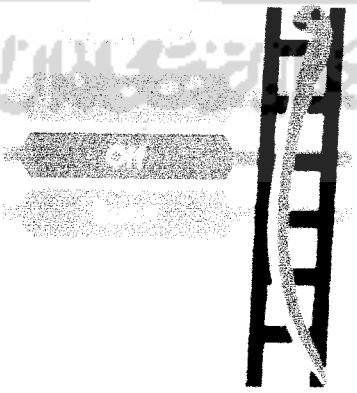
   Pada menu *Start Game* ini masukan yang dapat diterima adalah pada saat *user* menekan *keypad up*, *down* atau *center*. Contoh masukan yang dimasukkan untuk menguji keluaran output yang dihasilkan yaitu menekan *keypad up* sebanyak tiga kali. Hasil masukan pada menu utama tersebut dapat dilihat pada gambar 4.4



Gambar 4.4 Antar muka menu *Start Game* ketika *keypad up* ditekan tiga kali

3. Pada Menu *Sound*

   Pada menu *Sound* ini masukan yang dapat diterima adalah pada saat *user* menekan *keypad up*, *down* atau *center*. Contoh masukan yang dimasukkan untuk menguji keluaran output yang dihasilkan yaitu menekan *keypad down* sebanyak satu kali lalu menekan *keypad center*. Hasil masukan pada menu utama tersebut dapat dilihat pada gambar 4.5



Gambar 4.5 Antar muka menu *Sound* ketika *keypad up* ditekan satu kali lalu *keypad center* ditekan

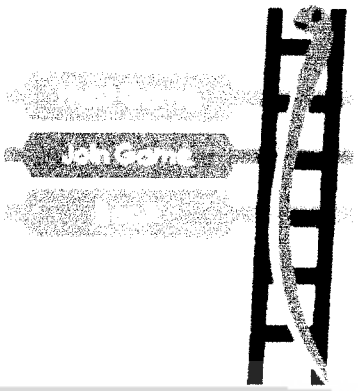
4. Pada Menu *Skins*

   Pada menu *Skins* ini masukan yang dapat diterima adalah pada saat *user* menekan *keypad up*, *down* atau *center*. Contoh masukan yang dimasukkan untuk menguji keluaran output yang dihasilkan yaitu menekan *keypad down* sebanyak tiga kali lalu menekan *keypad center*. Hasil masukan pada menu utama tersebut dapat dilihat pada gambar 4.6

Gambar 4.6 Antar muka menu *Skins* ketika *keypad down* ditekan tiga kali lalu *keypad center* ditekan

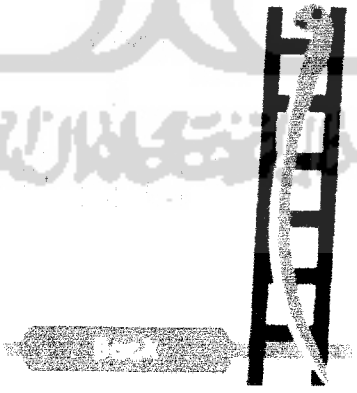
5. Pada Menu 2 *Players*

   Pada menu 2 *Players* ini masukan yang dapat diterima adalah pada saat *user* menekan *keypad up*, *down* atau *center*. Contoh masukan yang dimasukkan untuk menguji keluaran output yang dihasilkan yaitu menekan *keypad down* sebanyak satu kali. Hasil masukan pada menu utama tersebut dapat dilihat pada gambar 4.7



Gambar 4.7 Antar muka menu 2 *Players* ketika *keypad down* ditekan satu kali

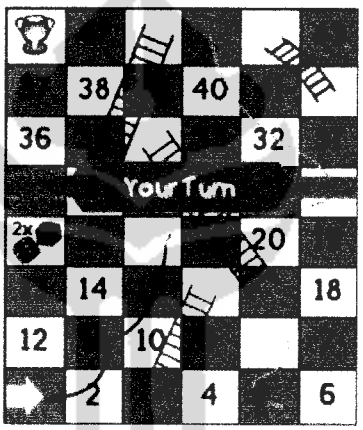
6. Pada halaman *Set Volume* ini masukan yang dapat diterima adalah pada saat *user* menekan *keypad right*, *left* atau *center*. Contoh masukan yang dimasukkan untuk menguji keluaran output yang dihasilkan yaitu menekan *keypad left* sebanyak tiga. Hasil masukan pada menu utama tersebut dapat dilihat pada gambar 4.8



Gambar 4.8 Antar muka halaman *Set Volume* ketika *keypad left* ditekan tiga kali

7. Pada saat bidak dijalankan

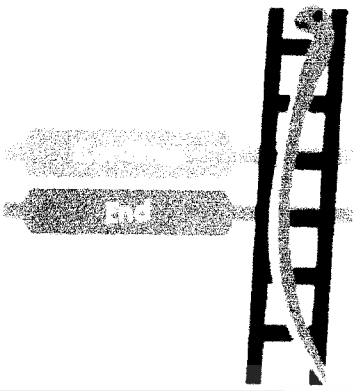
   Bidak dijalankan sesuai dengan angka dadu yang muncul. Untuk memunculkan angka dadu pertama-tama harus mengocok dadu terlebih dahulu dengan cara menekan *keypad center*, kemudian untuk menghentikan kocokan dadu, dengan cara menekan kembali *keypad center*. Contoh berhentinya bidak apabila angka dadu yang muncul adalah "5". Hasil jalannya bidak dapat dilihat pada gambar 4.9



Gambar 4.9 Antar muka Papan Permainan ketika bidak dijalankan lima langkah

8. Pada halaman Konfirmasi Melanjutkan Permainan

   Pada halaman konfirmasi melanjutkan permainan ini masukan yang dapat diterima adalah pada saat *user* menekan *keypad up*, *down* atau *center*. Contoh masukan yang dimasukkan untuk menguji keluaran output yang dihasilkan yaitu menekan *keypad down* sebanyak satu kali. Hasil masukan pada menu utama tersebut dapat dilihat pada gambar 4.10



Gambar 4.10 Antar muka halaman Konfirmasi Melanjutkan Permainan ketika *keypad down* ditekan satu kali

BAB V

SIMPULAN DAN SARAN

5.1 Simpulan

Berdasarkan hasil penelitian dan pembahasan yang telah dilakukan, dapat disimpulkan bahwa :

1. *Bluetooth* dapat dimanfaatkan dengan baik pada aplikasi permainan ular tangga ini untuk memudahkan permainan *multi player*.
2. *Mobile device* harus memiliki *memory* minimal 451 kB untuk dapat menggunakan aplikasi ini. Namun pada umumnya *smart phone* telah memiliki *memory external* yang cukup besar, sehingga tidak ada masalah dalam hal ini.

5.2 Saran

Mengingat berbagai keterbatasan yang dialami penulis terutama masalah pemikiran dan waktu, maka penulis menyarankan untuk pengembangan penelitian dimasa yang akan dating sebagai berikut :

1. Sebaiknya aplikasi permainan ular tangga ini dapat dimainkan oleh lebih dari dua orang pemain.
2. Pada papan permainan ular tangga, sebaiknya letak ular dan tangga dapat diacak, sehingga papan permainan menjadi lebih dinamis dan tidak membosankan.

DAFTAR PUSTAKA

[ANA05] Ananta, A.Y. 2005. Membangun Permainan Ular Tangga Berbasis-kan Multimedia. *Skripsi,* tidak diterbitkan. Yogyakarta : Fakultas Teknologi Industri Universitas Islam Indonesia.

[APR06] Apriany, D.S. 2006. Rancang Bangun Aplikasi Game Jaringan Pada Ponsel Berbasis Bluetooth Menggunakan Teknologi MIDP 2.0 dan CLDC 1.1. *Skripsi,* tidak diterbitkan. Yogyakarta : Fakultas Teknologi Industri Universitas Islam Indonesia.

[HAR04] Hartanto, A. A. 2004. *Pemrograman Mobile Java dengan MIDP 2.0,* Yogyakarta : Penerbit ANDI.

[SUY05] Suyoto, Dr. 2005. *Membuat Sendiri Aplikasi Ponsel,* Yogyakarta : Penerbit Gava Media.

[WAN05] Wang, A. I., Norum, M. S., & Lund, C. W., 2005. *Issues related to Development of Wireless Peer-to-Peer Games in J2ME* (On-line) Available at http://www.idi.ntnu.no

[WIC02] Wicaksono, A. 2002. *Pemrograman Aplikasi Wireless dengan Java,* Jakarta : Penerbit PT Elex Media Komputindo.

[XXX03] Forum Nokia 2003. *Bluetooth Technology Overview* (On-line) Available at http://www.forum.nokia.com

[XXX04] Developers Training Material 2004. *Developing Applications with the Java APIs for Bluetooth™ (JSR-82)* (On-line) Available at http://developer.sonyericsson.com/

[XXX07] Wikipedia Indonesia 2007. *Ular Tangga* (On-line) Available at http://id.wikipedia.org