IV. HASIL DAN PEMBAHASAN

A. Pengembangan Perangkat Lunak Berdasarkan RUP

Rancung bangun sistem permainan game *puzzle* pada *handphone* berbasis java menggunakan metode orientasi objek dan dibantu dengan menggunakan pemodelan UML. Proses pembuatan sistem dilakukan melalui sebuah proses yang dikenal dengan RUP (*Rational Unified Process*). Pada setiap fase dari RUP ini dihasilkan beberapa produk yang berhubungan dengan sistem permainan game *puzzle* pada *handphone* berbasis java. Fase-fase tersebut adalah fase insepsi, elaborasi, konstruksi, dan transisi. Pada setiap fase, di awal maupun di akhir selalu dilakukan evaluasi yaitu berupa perbaikan-perbaikan sistem dan penambahan fitur-fitur baru yang di dalam RUP itu sendiri dikenal dengan istilah iterasi dan inkrementasi.

1. Insepsi

Pada fase ini peneliti mulai melakukan analisis tentang gambaran produk secara umum, yang meliputi sasaran utama sistem (visi produk), batasan-batasan kerja sistem (*scope*), deskripsi umum sistem atau alur kerja secara umum, daftar aktor yang diperlukan beserta fitur masing-masing aktor (*user requirements*), dan hal-hal lainnya yang masih bersifat umum serta memiliki kemungkinan untuk berubah. Selain itu, peneliti juga mulai menentukan peralatan (*tools*) apa saja yang diperlukan untuk mendukung proses pembangunan sistem yang meliputi pemilihan perangkat lunak

a. Visi dan *Scope*

Penentuan visi dan *scope* ini sangat penting sebagai pengarah kerja dan batasan pembangunan sistem. Adapun visi dari permainan game *puzzle* pada *handphone* berbasis Java adalah :

1. Sebagai sarana game alternatif pada *handphone* berbasis java
2. Sebagai sarana untuk mengisi waktu luang.

Secara garis besar, visi ini sudah merupakan suatu batasan kerja yang dapat dilakukan oleh sistem, atau disebut juga dengan *scope*. Wilayah kerja sistem tidak mencakup tentang bagaimana menjalankan program game ini pada *handphone* yang memiliki sistem operasi selain java

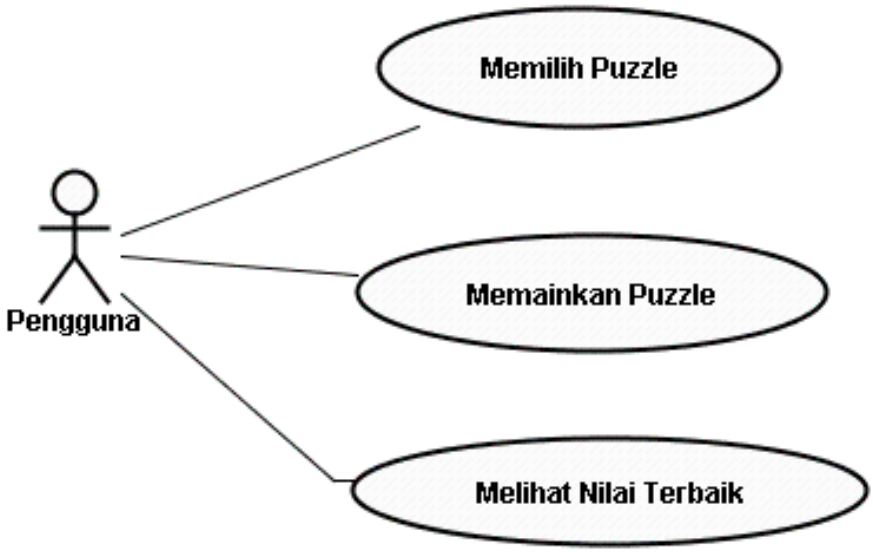
b. Deskripsi Umum Sistem

Aplikasi permainan game *puzzle* pada *handphone* berbasis java adalah suatu sistem aplikasi yang dipakai sebagai sarana untuk menjalankan program *game* pada ponsel dengan fungsi utama sebagai sarana *game* alternatif sehingga dapat menjadi alternatif permainan dan hiburan bagi para pengguna ponsel.

Pada aplikasi ini hanya terdapat satu orang aktor yaitu *user* atau pengguna ponsel itu sendiri, yang mana pengguna tersebut dapat mengakses dan menggunakan fitur-fitur yang ada pada aplikasi ini.

c. General Use case

Produk *general use case* ini berisi gambaran fungsi-fungsi dari aktor secara umum. Penjelasan masih bersifat umum dan memiliki kemungkinan untuk berubah.



Gambar 15**.** *General use case*

Pada diagram *use case* diats terdapat gambaran keseluruhan pengguna sistem. *Use case* ini bersifat umum maksudnya yaitu bias digunakan oleh semua pengguna yaitu memilih *puzzle*, memainkan *puzzle*, melihat nilai terbaik.

d. Identifikasi Resiko

Tabel 7. Identifikasi Resiko Pembangunan Sistem

Tingkat	Deskripsi	Akibat	Strategi & Antisipasi
1	Kurang menguasai bahasa pemrograman *Java* dengan model OOP	Kode program tidak terstruktur, sulit untuk dikembangkan	Membaca literatur dan mempelajari kembali dari buku dan tutorial tentang *Java* dan OOP
2	Terjadi kesalahan pada saat pembuatan dan penginstalan aplikasi pada ponsel	Ponsel dan program yang sudah ada akan mengalami kerusakan atau kesalahan	Mempelajari kembali arsitektur dan detail dari ponsel tersebut dan sistem operasi yang digunakannya.
3	Kurang mengetahui kebutuhan pihak user.	Tidak lengkapnya fitur-fitur yang ada dalam sistem	Sering melakukan koordinasi dan survey dengan *user* (pengguna ponsel)
4	Ada beberapa fitur yang mungkin masih sulit untuk diimplementasikan	Fungsi kerja sistem tidak dapat digunakan seluruhnya	Mencari kode program yang berhubungan dengan fitur tersebut (dari *manual book* atau internet) atau dapat juga mencari alternatif fitur lain.
5	Kapasitas ruang penyimpanan yang ada pada ponsel tidak terlalu besar.	Jenis dan macam kata yang ada pada *record store* menjadi sedikit	Mencari cara untuk mengkompressi file suara tersebut agar kapasitasnya menjadi lebih kecil.

e. Membangun Lingkungan yang Mendukung dan Infrastruktur yang Digunakan

Dalam proses pembuatan aplikasi permainan puzzle pada *handphone* berbasis Java ini, digunakan beberapa *software* yaitu :

1. Java SE *Development Kit* 6 sebagai bahasa pemrograman

2. Java *Micro Edition* sebagai bahasa pemrograman untuk ponsel
3. J2ME *Wireless Toolkit* sebagai *emulator* ponsel
4. NetBeans IDE 6.5 sebagai editor *java*
5. JUDE sebagai *software* pemodelan untuk UML

Software-software tersebut diinstalasi dan dikonfigurasi sehingga dapat digunakan.

f. Kandidat Arsitektur Sistem

Dalam membuat dan membangun sebuah sistem diperlukan arsitektur yang sesuai dengan cara kerja dari sistem yang akan dibuat tersebut. Pada gambar di bawah ditunjukkan suatu model arsitektur sistem, di mana pengguna *(user)* berhubungan langsung dengan aplikasi (*application*). Untuk memperoleh data, pengguna tidak langsung terhubung tetapi melalui perantara dari aplikasi yang akan mengambil data dan memberikannya ke pengguna.

Gambar 16. Model Arsitektur Sistem

2. Fase Elaborasi

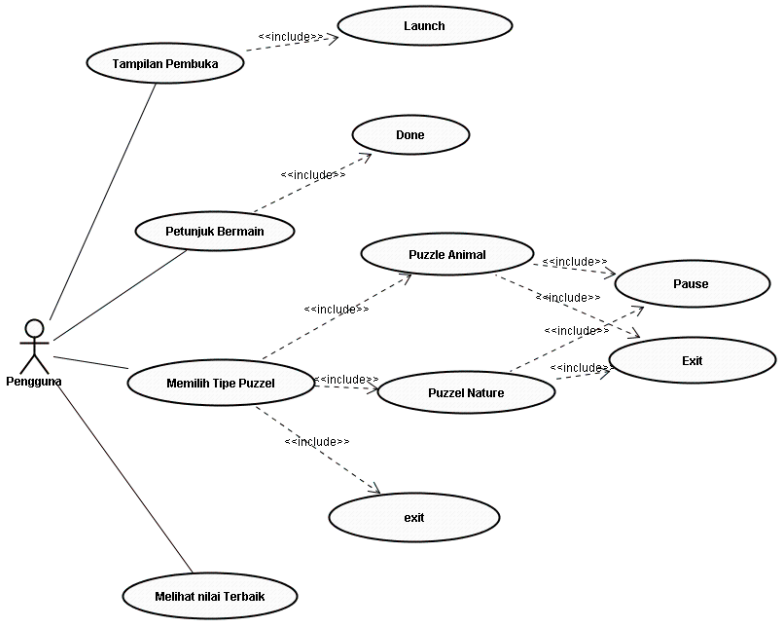
Pada fase elaborasi, peneliti mulai menganalisa kebutuhan aplikasi secara rinci dan mulai mendefinisikan pondasi arsitektur sistem. Langkah ini berfungsi untuk mengurangi resiko kesalahan teknis dalam pembangunan sistem. Di antaranya dengan melakukan analisis *use case* lebih detail, membuat skenario *use case,* membuat desain kelas dan menentukan jenis relasi antar kelas, atribut dan method yang diperlukan secara umum berdasarkan analisis *user requirements* yang telah dibuat pada fase insepsi. Kemudian membuat desain relasi *database* berdasarkan desain kelas yang telah dibuat, dan menentukan arsitektur sistem secara keseluruhan. Meskipun demikian, proses perbaikan dan penambahan hasil analisis di fase sebelumnya tetap dilakukan.

a. Melakukan Identifikasi *Use Case*

Tabel 8. Identifikasi *Use Case*

No.	***Requirement***	***Use Case***	**Deskripsi**
1	Memilih Permainan	*Play Puzzle*	Pengguna masuk ke dalam sistem untuk memulai permainan.
2	Memilih tipe *Puzzle*	*Choose Puzzle*	Pengguna memilih tipe puzzle apa yang ingin dimainkan.
3	Memilih Animal Puzzle	*Animal Puzzle*	Pengguna memilih puzzle animal
4	Memilih Nature Puzzle	*Nature Puzzle*	Pengguna memilih puzzle nature
5	Memulai Permainan	*Play Puzzle*	Pengguna memulai permainan Puzzle nya
6	Keluar permainan	*Quit Puzzle*	Pengguna keluar dari sistem
7	Melihat nilai terbaik	*Best Exchange*	Melihat nilai pertukaran terbaik

b. Detail *use case*



Gambar 17. Detail Use Case

Pada diagram *use case* terdapat fungsi memulai permainan yang berfungsi untuk kita masuk ke dalam sistem untuk memulai permainan puzzle. Pada *use case* Memilih tipe puzzle menggambarkan bahwa *use case* Animal puzzle dan Nature puzzle merupakan fungsional dari use case memilih tipe puzzle, di mana setelah kita memilih menu tipe puzzle kita akan dapat jenis permainan puzzle yang akan kita mainkan. Selanjutnya pada diagram *use case* tampilan pembuka, menu *Pause* dan *Exit* juga merupakan fungsional dari *use case* memulai permainan.

c. Skenario *Use case*

Skenario *use case* berisi tentang gambaran *use case* secara mendetil. Di dalamnya terdapat nama *use case*, deskripsi, aktor, pemicu, kondisi sebelumnya, alur kejadian, pengecualian dan kondisi sesudahnya yang digambarkan dalam bentuk tabel. Untuk tabel skenario *use case* dapat dilihat pada lampiran.

d. Desain *Class Diagram*

Untuk menjembatani proses analisis dengan proses disain perlu disisipkan satu aktivitas untuk mengidentifikasikan *class* yang terlibat dari spekisifikasi *use case* atau biasa disebut analisis *use case* yang meliputi:

1. Dari setiap use case diidentifikasi *class* yang terlibat.
2. Dari setiap *class* yang bisa diidentifikasi ditentukan atributnya.

e. Identifikasi *Class*

Dilakukan untuk mengetahui atribut-atribut apa yang ada pada *class* yang akan digunakan serta menjelaskan fungsi-fungsi pada operasi yang ada pada *class* yang akan digunakan tersebut.

Tabel 9. Identifikasi *Class*

No.	Nama *Class*	Atribut	Operasi
1	Mobile Device	*a. Started* Memulai permainan b. Resumed Melanjutkani permainan	a. *startMIDlet()* *Method* yang berfungsi untuk memulai aplikasi game b. *resumeMIDlet* () *Method* yang berfungsi untuk melanjutkan permainan
2	*Main Menu*	*a. Exit* untuk keluar dari permainan **b.** *Launch* Memulai permainan	a. getExitCommand() *Method* yang berfungsi untuk keluar dari permainan b. getLaunchCommand() *Method* yang berfungsi untuk memulai game
3	*Puzzle1*	a. *Play* Memulai permainan b. *Back* kembali ke menu sebelumnya	a. getPlayCommand() b. getBackCommand()
4	*Puzzle2*	a. *Play* Memulai permainan b. *Back* kembali ke menu sebelumnya	a. getPlayCommand() b. getBackCommand()
5	frmHasil	a. Back	a. getBackCommand()

Relasi antar *class* pada aplikasi ini merupakan relasi asosiasi. Relasi asosiasi terdapat pada *class-class* yang memiliki hubungan relasi *one-to-many* maupun *many-to-many*, namun tidak memiliki ketergantungan yang erat dengan *class* yang dituju. Contohnya pada *Main Menu* dengan *Puzzel1* memiliki hubungan relasi di mana *Class Main Menu* tidak tergantung secara erat dengan kelas *Puzzel1*, bila objek pada *class Puzzel1* dihapus atau dihilangkan, maka tidak akan menghapus objek dari *class Main Menu*. Untuk lebih jelasnya, gambar *class* diagram aplikasi dapat dilihat pada Lampiran E.

f. Desain *Sequence Diagram*

Menggambarkan interaksi antar objek di dalam sistem berupa pesan yang digambarkan terhadap waktu. Fungsinya untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah kejadian untuk menghasilkan keluaran tertentu. Pada *sequence diagram* sistem ini akan digambarkan interaksi antar objek yang diawali dari apa yang memicu aktivitas tersebut, proses dan perubahan apa saja yang terjadi di dalam sistem dan keluaran apa yang dihasilkan. *Sequence diagram* lebih detil dapat dilihat pada Lampiran D.

g. Desain *Activity Diagram*

Menggambarkan rangkaian aliran dari aktivitas dari sistem yang dirancang, bagaimana masing-masing aliran berawal, hasil yang mungkin dan bagaimana mereka berakhir.

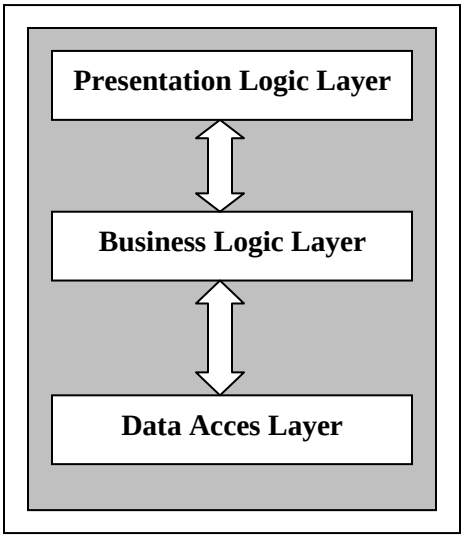
Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* lebih detil dapat dilihat di Lampiran C.

h. Arsitektur *Framework*

Pada aplikasi ini, arsitektur sistem terdiri atas tiga tingkat atau disebut juga *three tier architecture*. Sistem dibagi menjadi tiga bagian komponen yang terpisah terdiri atas:

1. Bagian akses data (*data acces*)
2. Bagian logika bisnis (*business logic*)
3. Bagian presentasi (*presentation*)

Lapisan akses data merupakan implementasi dari manajemen *database* sistem dan integrasinya ke dalam sistem. Di dalam lapisan ini berisi hal-hal yang berhubungan dengan *database*. Pada lapisan logika bisnis mempunyai peran sebagai logika dan alur kerja sistem yang menghubungkan antara *presentation* dengan dengan *data acces*.



Gambar 18. Arsitektur Tiga Tingkat

Sedangkan presentasi yang merupakan bagian untuk mengatur tampilan halaman yang dilihat oleh pengguna. Pada bagian ini pengguna dapat memilih menu yang tersedia di mana fungsi dari menu diatur oleh bagian logika bisnis.

3. Konstruksi

Tujuan utama dari fase konstruksi ini adalah menstabilkan produk agar dapat di integrasikan ke lingkungan pengguna dan menjamin bahwa lingkungan pengguna menerima produk yang dihasilkan tersebut. Adapun produk-produk yang dihasilkan di dalam fase konstruksi yaitu :

a. Pengujian *Use Case*

Tahap ini berisi pengujian terhadap *use case-use case* yang ada. Pengujian ini bertujuan untuk mengetahui apakah *use case-use case* yang sudah ada berjalan atau tidak dan apakah *use case* tersebut sesuai atau tidak dengan skenario *use case* yang telah dibuat pada fase elaborasi.

Gambar 19. Pengujian *Use Case* ZPUZZLE pada emulator

Pada pengujian *Use Case ZPUZZLE* ini dapat dilihat pada gambar di atas bahwa pengguna harus memilih menu *Launch* untuk memulai permainan. Setelah itu aplikasi akan menampilkan gambar tampilan splash Puzzle, kita dapat memilih menu keluar atau main. Jika memilih menu main maka setelahnya akan tampil menu penunjukan cara permainan dan pemilihan tipe permainan

Gambar 20. Pengujian Petunjuk Cara Bermain ZPUZZLE pada Emulator

Pengujian *Use Case* selanjutnya adalah pengujian proses pengacakan Puzzel,baik itu proses pengacakan Puzzel Animal maupun Puzzel Nature,serta fungsi-fungsi tombol pada permainan, setelah memilih menu **done** maka kita akan langsung memulai permainan seperti di bawah ini

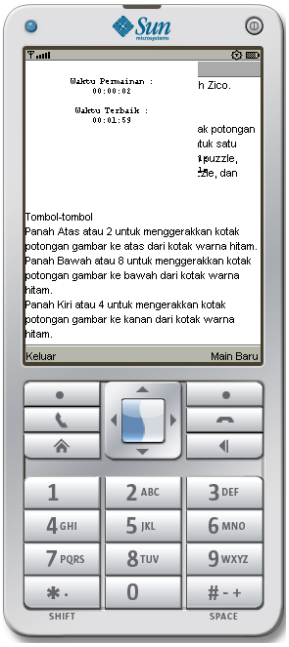
Gambar 21. Gambar *Puzzle Animal* dan *Puzzle Nature* pada Emulator

Pada pengujian di atas fungsi tombol akan mengikuti posisi kotak hitam, jadi jika kita menekan tombol atas maka kotak hitam akan bergeser ke atas, jika kita menekan tombol kiri maka kotak hitam akan bergeser ke kiri, begitu juga dengan arah-arah lainnya

```
public SplashScreen getPuzzle1() {
    if (puzzle1 == null) {
        // write pre-init user code here
        puzzle1 = new SplashScreen(getDisplay());
        puzzle1.setTitle("Puzzle Animal");
        puzzle1.addCommand(getPlayCommand());
        puzzle1.addCommand(getBackCommand());
        puzzle1.setCommandListener(this);
        puzzle1.setImage(getImage2());
        puzzle1.setTimeout(Integer.MAX_VALUE);
        // write post-init user code here
    }
    return puzzle1;
}
```

Gambar 22. Potongan Kode Program Animal

Setelah menyelesaikan permainan *Puzzle*, pengguna dapat melihat hasil permainan, dimana akan terlihat nilai terbaik yang tersimpan sebelumnya dan nilai kita sendiri. Pada tampilan juga terdapat pilihan back untuk kembali ke menu selanjutnya. Lebih jelasnya pengujian *Use Case* hasil permainan dapat kita lihat pada gambar di bawah ini



Gambar 23. *Use Case* hasil permainan pada Emulator

Tabel 10. Pengujian *Use Case*

NO	PENGUJIAN *USE CASE*	AKSI	HASIL
1	*ZPUZZLE*	*Launch*	Menampilkan Menu Pembuka
2	Menu Pembuka	Mulai	Menampilkan Menu Petunjuk
		keluar	Kembali Ke Halaman Awal
3	Menu Petunjuk	*Done*	Menu Pemilihan *Puzzle*
4	Menu Pemilihan *Puzzle*	Nomor 1	Permainan *Puzzle Animal*
		Nomor 2	Permainan *Puzzle Nature*
		Exit	Kembali ke Halaman Awal
5	*Puzzle Animal* Dan *Nature*	Tunda	Permainan Berhenti
		Keluar	Kembali Ke Halaman Awal
6	Hasil Permainan	Main Baru	Kembali Ke Halaman Pemilihan
		Keluar	Kembali Ke Halaman Awal

b. Pengujian *Beta Release*

Setelah dipastikan bahwa *use case* berjalan dengan baik, maka dilakukan pengujian terhadap *use case* secara keseluruhan dan mengujinya dengan memainkan permainan Puzzel pada *HandPhone* berbasis java

4. Transisi

Setelah *melakukan* berbagai penyempurnaan dan perbaikan, maka sistem siap digunakan dan disalurkan kepada pengguna. Tetapi sebelumnya dilakukan pengujian secara menyeluruh terlebih dahulu oleh penguji sistem.

a. Pengujian Aplikasi (*Final Release*)

Pada tanggal 11 November 2010, dilakukan pengujian secara keseluruhan. Pengujian dilakukan dengan memainkan aplikasi permainan puzzle pada *handphone* dan juga mencoba semua fitur yang tersedia serta tidak lupa mencoba aplikasi pada beberapa ponsel yang berbeda tipe. Dari hasil pengujian didapatkan hasil yang cukup memuaskan, walaupun ada juga hasil yang kurang memuaskan. Hal tersebut memang dapat terjadi dikarenakan tiap-tiap *handphone* memiliki resolusi layar yang berbeda-beda.

b. Packaging *ZPUZZLE*

Tahap selanjutnya adalah mengemas perangkat lunak aplikasi yang diberi nama ZPUZZLE. Pengepakan dilakukan dengan mengubah kode program menjadi format .jar dengan ukuran sekitar 135 KB. Dengan hasil ini maka dapat disimpulkan bahwa sistem sudah dapat dipergunakan oleh kalangan pengguna.

c. Hasil Pengujian Perangkat Lunak

Untuk menguji peerangkat lunak, maka penulis melakukan penelitian dengan melakukan instalasi perangkat lunak pada beberapa jenis dan tipe ponsel yang berbeda. Pengujian yang dilakukan menghasilkan informasi tingkat keberhasilan penginstalan perangkat lunak yang dibuat dan jalannya perangkat lunak setelah dimasukkan ke dalam ponsel. Berdasarkan pengujian yang telah dilakukan, maka didapatkan hasil pada tabel sebagai berikut:

Tabel 11. Hasil Pengujian Perangkat Lunak

NO	PONSEL	HASIL
1	Nokia 7610	Dapat Dimainkan
2	Nokia E73	Dapat Dimainkan
3	Nokia 6120 Klasik	Dapat dimainkan
4	Blackberry 9000	Dapat Dimainkan
5	Blackberry 8520	Dapat Dimainkan
6	Nokia 1200	Tidak Bisa Dimainkan

Pada table di atas terdapat enam jenis ponsel yang digunakan untuk pengujian perangkat lunak, sebenarnya masih banyak tipe ponsel lainnya di pasaran, tetapi di sini peneliti hanya mengujikan pada enam ponsel saja karena keterbatasannya jumlah ponsel yang Bisa Diakses.

Dari pengujian, dari enam ponsel lima ponsel dapat memainkan Puzzel dan hanya satu ponsel yang tidak bisa, bila dipersentasekan maka tingkat keberhasilan pengujian perangkat lunak adalah 83,33%.

B. Pembahasan

1. Pengaturan Pada Ponsel

Pada saat pengaturan resolusi, peneliti mengalami kesulitan, karena ponsel 7610 yang di spesifikasi pabrikannya disebutkan memiliki resolusi 176 x 208 ternyata hanya support 176 x 140, sedangkan emulator yang tersedia di Netbeans rata-rata mendukung 240 x 320 ke atas, jadi peneliti harus menyesuaikan langsung di ponsel nya. Selain itu karena ponsel nokia 7610 masih menggunakan symbian s60, tidak bisa menampilkan Puzzel dalam bentuk *full screen*, sehingga tampilan Puzzel nya hanya setengah layar seperti gambar 21.

Kemudian karena pembuatan Puzzel ini menggunaka fungsi *graphics* java, maka gambarnya harus dibuat di *canvas* bukan di *form*. Selain itu karena ponsel Nokia 7610 ini hanya *support* CLDC 1.0, alokasi *pixel* gambar serta fungsi-fungsinya terbatas, sehingga penulis mengalami kesulitan dalam membuat algoritma *control*-nya.

2. Keberhasilan Perangkat Lunak

Selama percobaan didapatkan bahwa semua Ponsel berbasis Java maupun Symbian yang mendukung untuk program Java dapat menjalankan atau menginstalasi program ini. Sedangkan ponsel yang tidak berbasis Java tidak bisa menjalankan program ini. Hal tersebut dapat dilihat pada Tabel pengujian.

Walaupun semua nya dapat menjalankan program, tiap-tiap ponsel menampilkan tampilan yang tidak sama dalam ketajaman gambarnya, hal ini disebabkan tiap-tiap ponsel memiliki resolusi yang berbeda. Selain itu tiap-tiap ponsel juga memiliki CLDC yang berbeda.

Kemudian keterbatasan pengetahuan peneliti tentang pemrograman Java menyebabakan tidak semua ponsel menghasilkan tampilan yang sama, dan kontrol permainan yang belum sempurna. Pengendalian seharusnya disesuaikan dengan posisi gambar tetapi pada permainan ini kontrolnya disesuaikan dengan posisi kotak hitam.

Namun telah disampaikan oleh peneliti di bab awal bahwa aplikasi yang dibuat memiliki kekurangan dibandingkan aplikasi yang sudah ada dan dibuat sebelumnya. Tapi disini penulis juga memberikan nilai tambah dengan memberikan petunjuk cara permainan dan pilihan gambar yang lebih dari satu. Penulis sudah berupaya dengan semaksimal mungkin untuk membuat aplikasi yang akurat dan tepat serta sesuai dengan penggunaan dan tujuan awal dari pembuatan aplikasi ini.