BAB 1
PENDAHULUAN

1.1 Latar Belakang

Dalam perkembangannya, diagram UML yang biasanya digunakan untuk menggambarkan desain sistem juga dapat digunakan untuk pengujian atau biasa disebut *model-based testing* (MBT). Jika dibandingkan dengan *traditional testing* yang harus menunggu suatu perangkat lunak selesai dibangun untuk melakukan *testing,* MBT lebih unggul dalam segi waktu karena menghilangkan beberapa proses pada *traditional testing* dihilangkan.

Dalam *model-based testing*, UML digunakan pada pembangkitan *test case* yang nantinya akan dijalankan ke *system under test* (SUT). Hambatan dalam proses membangkitkan *test case* adalah apabila diketahui model yang digunakan rumit atau terlalu besar. Dibutuhkan waktu yang banyak untuk membaca dan mengubah model satu per satu ke sekumpulan *test case*.

Sedangkan untuk membangkitkan *test case* hanya diperlukan satu atau beberapa diagram saja, misalnya *activity diagram*. *Activity diagram* adalah representasi grafis dari seluruh tahapan alur kerja. Keunggulan dipilihnya *activity diagram* adalah diagram ini menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Alasan tersebut mendasari penelitian ini menggunakan *activity diagram* sebagai data masukannya. Tetapi ada masalah lain dalam dipilih *activity diagram*, yaitu seringkali *developer* tidak menggambarkan *activity diagram* dalam notasi yang lengkap, hanya menggunakan notasi dasar saja karena beberapa informasi desain sudah tergambarkan dalam diagram UML lainnya.

Masalah lain muncul dalam pencarian *test path* suatu *activity diagram*. *Test path* yang berarti jalur tes yang menjadi cikal bakal *test case*. Dalam penelitian sebelumnya, yaitu oleh Debasish Kundu dan Debasis Samanta [6] dominan digunakan algoritma pencarian *depth first search* yang mempunyai *test coverage* kurang rinci. *Test coverage* merupakan cakupan algoritma dalam menangani suatu masalah. Sehingga digunakanlah *beardth first search* yang notabene mempunyai cakupan lebih rinci [11].

Munculnya beberapa permasalahan diatas, mengilhami dilakukan penelitian untuk bagaimana membuat suatu *tools* yang dapat membantu membangkitkan *test case* menggunakan *activity* diagram dengan didasari metode *model-based* testing. Maka diangkatlah tugas akhir dengan judul "*Model-Based Test Case Generation* Menggunakan *Activity Diagram*".

1.2 Perumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas maka dapat dirumuskan suatu permasalahan, yaitu:

1. Bagaimana membuat proses automasi *model-based test case generation* menggunakan *activity diagram*?
2. Apakah *test case* yang dihasilkan dari perumusan masalah nomer 1 sudah sesuai?
3. Apakah diperlukan proses verifikasi untuk *activity diagram* sebelum masuk ke sistem?
4. Bagaimana *test case* saat dijalankan pada *system under test*?

1.3 Tujuan

Tujuan yang ingin dicapai dalam Tugas Akhir ini adalah:

1. Membangun sebuah *tools model-based test case generator* menggunakan *activity diagram*.
2. Melakukan analisis kesesuaian *test case* (*expected result=actual result)* yang dihasilkan dari *tools* yang telah dihasilkan di nomer 1.
3. Melakukan analisis terhadap *activity diagram* sesuai kebutuhan sistem.
4. Melakukan *testing* pada *system under test*.

1.4 Batasan Masalah

Batasan – batasan yang diteliti agar tidak meluasnya pembahasan, sebagai berikut:

1. Studi kasus yang digunakan adalah perangkat lunak ATM Simulation dan Website Bandung Pintar yang sudah disertai dokumentasinya.
2. Menggunakan Selenium untuk simulasi pengujian website, sedangkan pengujian perangkat lunak desktop dilakukan secara manual.
3. Pembuatan *activity diagram* dan *generate activity diagram* ke XMI menggunakan IBM Rasional Software Architect.
4. Tidak ada perubahan di setiap proses/aktivitas pada *activity diagram* (diasumsikan benar).
5. Penelitian tidak memperhitungkan penggunaan rangka kerja/*framework* yang digunakan pada studi kasus.

1.5 Metodologi Penyelesaian Masalah

Metodologi yang dipakai dalam tugas akhir ini adalah:

1. Studi Literatur
   Pada tahap ini dipelajari mengenai pemahaman konsep aplikasi berbasis objek, *software testing, model-based testing, UML, activity diagram, XMI, bfs*. Literatur yang digunakan diperoleh dari internet berupa makalah ilmiah, tesis, serta beberapa buku referensi.

2. Pengumpulan data
   Mengumpulkan data-data yang diperlukan untuk keperluan tugas akhir. Pada tugas akhir ini data-data yang dikumpulkan adalah dokumen SRS yang mempunyai *activity diagram* yaitu diperoleh dokumen SRS website Bandung Pintar [7] dan ATM Simulation [4] . Dari dokumen tersebut diperoleh *activity diagram* yang dapat digunakan untuk membangkitkan sebuah *test case*.

3. Perancangan dan Implementasi Sistem
   Pada tahap ini dirancanglah sistem yang akan dibangun dengan menganalisis kebutuhan sistem seperti bagaimana parsing *activity diagram* XMI, bagaimana membangun *activity graph*, dan bagaimana mencari *test path* yang kemudian diperoleh sekumpulan *test case*. Untuk pemilihan algoritma pencarian *test path* digunakan algoritma *breadth first search* setelah dilakukan studi literatur yang mendalam. Selanjutnya dilakukan pembangunan sebuah *tools* yang sebelumnya telah dirancang dengan bahasa pemrograman java.

4. Pengujian dan Analisis Hasil
   Pengujian dilakukan untuk mengetahui kehandalan dari *tools* yang telah dibangun. Setelah aplikasi yang dihasilkan telah diuji, selanjutnya adalah dijalankan pada *system under test* pada website Bandung Pintar dan Simulasi ATM.

5. Pengambilan Kesimpulan dan Pembuatan Laporan
   Menyimpulkan hasil analisis dan kemudian mendokumentasikan menjadi sebuah buku. Dokumentasi ini dibuat agar memudahkan peneliti lain yang ingin mengembangkan metode atau aplikasi ini lebih lanjut.

1.6 Sistematika Penulisan

Tugas Akhir ini disusun menjadi lima bab, dengan rincian sebagai berikut:

BAB I PENDAHULUAN
Bab ini membahas tentang latar belakang, perumusan masalah, tujuan, batasan masalah, metodologi serta sistematika penulisan dari tugas akhir ini.

BAB II LANDASAN TEORI
Bab ini berisikan tentang teori yang mendukung dan mendasari penulisan tugas akhir ini, yaitu tentang *pengujian perangkat lunak*, pembangkitan *test case*, *activity diagram*, dan teori lainnya yang digunakan dalam penulisan tugas akhir ini.

BAB III PERANCANGAN SISTEM
Bab ini menguraikan tentang proses perancangan dalam mengimplementasikan *model-based test case generation* menggunakan *activity diagram*.

BAB IV PENGUJIAN DAN ANALISIS SISTEM
Bagian ini menampilkan skenario pengujian terhadap sistem yang telah dibangun dan menampilkan hasil pengujian ke studi kasus dengan *test case* yang telah dihasilkan oleh sistem.

BAB V KESIMPULAN DAN SARAN

Bagian ini merupakan kesimpulan dari hasil pengujian dan analisis sistem serta saran untuk perkembangan ke depannya mengenai pembangkitan *test case.*