Pengujian Berbasis *Behaviour Specification*

Muhammad Dipo Putra Wandara–2016730091

1 Data Skripsi

Pembimbing utama/tunggal: **Raymond Chandra Putra**
Pembimbing pendamping: -
Kode Topik : **RCP4702**
Topik ini sudah dikerjakan selama : **1** semester
Pengambilan pertama kali topik ini pada : Semester **47 - Ganjil 19/20**
Pengambilan pertama kali topik ini di kuliah : **Skripsi 1**
Tipe Laporan : **B -** Dokumen untuk reviewer pada presentasi dan **review Skripsi 1**

2 Latar Belakang

Pengujian perangkat lunak merupakan salah satu tahapan dari *software development life cycle*. *Software Development Life Cycle*(SDLC) merupakan metodologi dari pengembangan perangkat lunak, metode ini dibagi menjadi beberapa fase seperti *analysis, design, coding, testing, installation* dan *maintenance*[7]. *Testing*/Pengujian perangkat lunak adalah proses untuk mencari kesalahan pada setiap *item* perangkat lunak, mencatat hasilnya, mengevaluasi setiap aspek pada setiap komponen (sistem) dan mengevaluasi fasilitas-fasilitas dari perangkat lunak yang akan dikembangkan[1].Pengujian pada perangkat lunak merupakan tahapan yang wajib dilakukan sebelum perangkat lunak tersebut digunakan, agar memastikan sudah tidak ada *error/bug* pada perangkat lunak yang sedang dikembangkan. Pengujian perangkat lunak memakan *resource* yang berat, baik itu waktu maupun waktu maupun tenaga kerja, karena untuk melakukan pengujian perangkat lunak dibutuhkan *developer* ahli *Software Quality Assurance*(SQA). SQA yang bertanggung jawab untuk memastikan bahwa perangkat lunak yang sedang dikembangkan bebas dari *bug* agar siap untuk di-*release* dan digunakan oleh pengguna.

Unit testing merupakan tahapan pertama dari pengujian perangkat lunak. *Unit Testing* adalah proses pengujian bagian kode secara individu, suatu komponen, untuk menentukan apakah kode tersebut berfungsi dengan semestinya[5]. Salah satu teknik untuk membuat *unit testing* yaitu dengan code generation. *Code generation* adalah teknik membuat suatu program yang membuat program lain. Menggunakan *code generation* untuk *unit testing* membuatnya mudah untuk mempertahankan dan memperpanjang *unit testing*[6].

Ada beberapa metode untuk melakukan pengujian perangkat lunak, salah satunya yaitu *Test-Driven Development*(TDD). TDD adalah praktik pemrograman yang menginstruksikan pengembang perangkat lunak untuk menulis kode baru hanya ketika tes yang dilakukan secara otomatis gagal, dan menghilangkan duplikasi. Tujuan TDD adalah kode bersih yang berfungsi[2]. Tetapi, TDD memiliki masalah, bahwa TDD berfokus pada keadaan sistem daripada *behaviour* yang diinginkan oleh sistem, dan kode pada pengujian *highly coupled* dengan implementasi sistem yang ada[3]. Maka dari itu metode yang akan digunakan pada skripsi ini adalah *Behaviour-Driven Development*.

Behaviour-Driven Development(BDD) merupakan evolusi dari TDD untuk menyelesaikan masalah yang ada pada TDD. BDD adalah pendekatan pengembangan perangkat lunak yang *agile* untuk mendorong kolaborasi antara semua peserta dalam proses pengembangan perangkat lunak[4].

Prinsip inti dari BDD adalah "orang bisnis dan teknologi harus merujuk ke sistem yang sama dengan cara yang sama"[4]. Berbeda dengan TDD yang berpatokan pada test untuk *develope* lebih jauh program yang akan digunakan, BDD berpatokan pada keinginan *stakeholder/customer* untuk mengembangkan programnya. Untuk mencapai kesepakatan antara *developer* dan *stakeholder/customer*, maka dibutuhkan *language* untuk menspesifikasikan *behaviour* sebuah sistem agar kedua pihak paham, dan dapat mewujudkan hal berikut[4]:

- *Stakeholder/Customer* untuk menentukan persyaratan dari perspektif bisnis.
- Analis bisnis untuk melampirkan contoh konkret (skenario atau *acceptance tests*) yang menjelaskan *behaviour* sistem.
- *Developer* untuk mengimplementasikan *behaviour* sistem yang diperlukan secara TDD.

Pengujian dengan basis *behaviour specification* akan berfokus kepada hal yang *stakeholder/customer* harapkan pada suatu sistem dengan *behaviour* yang sudah di spesifikasikan. Pengujian akan berawal dengan membuat *Specification* yang berisi [4]:

1. *Context/Starting state* : Posisi awal sistem sebelum terjadinya suatu *event*.
2. *Event* : *Task* yang dilakukan oleh pengguna pada sistem.
3. *Outcome* : Hasil yang diharapkan dari sistem.

Developer akan menjadikan *specification* sebagai patokan dasar bekerjanya suatu sistem, dan akan melakukan pengujian berbasis *specification*. Pengujian dilakukan untuk memastikan bahwa sistem sudah bekerja sesuai apa yang *stakeholder/customer* harapkan, dan jika *specification* berhasil dilakukan, maka sistem sudah siap untuk digunakan.

Pada skripsi ini akan dibangun sebuah perangkat pengujian yang mengimplementasi behaviour specification untuk menguji sebuah perangkat lunak.

3 Rumusan Masalah

- Bagaimana cara kerja pengerjaan Pengujian Berbasis Behaviour Specification?
- Bagaimana implementasi perangkat lunak yang dapat menguji program berbasis Behaviour Specification ?

4 Tujuan

- Mempelajari cara kerja Pengujian Berbasis Behaviour Specification
- Menghasilkan perangkat penguji yang dapat menguji sebuah perangkat lunak berbasiskan *behaviour specification*.

5 Detail Perkembangan Pengerjaan Skripsi

Detail bagian pekerjaan skripsi sesuai dengan rencan kerja/laporan perkembangan terkahir :

1. **Studi literatur mengenai pengujian perangkat lunak, *unit testing*, dan *behaviour specification*.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Studi literatur telah dilakukan dengan membaca berbagai buku yang berkaitan dengan pengujian perangkat, *unit testing*, dan *behaviour specification*. Buku yang digunakan untuk referensi yaitu *Software Testing: A Craftsman's Approach, Fourth Edition*, *Software Testing and Continuous Quality Improvement, Third Edition*, *Testing Python: Applying Unit Testing, TDD, BDD and Acceptance Testing*, dan *BDD in Action: Behavior-Driven Development for the whole software lifecycle*

2. **Melakukan analisis pada hasil survei terhadap pergerakan pengunjung di museum dan membuat rancangan denah di komputer yang dilengkapi dengan penghalang dan objek di museum.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

3. **Melakukan studi literatur mengenai sifat kolektif suatu kerumunan, teknik** *social force model* **dan teknik** *flow tiles*
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

4. **Mempelajari bahasa pemrograman C++ dan cara menggunakan framework OpenSteer**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

5. **Merancang pergerakan kerumunan di dalam museum menggunakan teknik** *social force model* **dan** *flow tiles* **serta menggunakan teknik lainnya seperti konsep pathway dan waypoints. Selain itu, dirancang pula adanya waktu tunggu (pada saat pengunjung melihat objek di museum) dan cara pembuatan jalur bagi setiap individu pengunjung**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

6. **Melakukan analisa dan merancang struktur data yang cocok untuk menyimpan penghalang (obstacle)**
   Status : dihapuskan/tidak dikerjakan
   Hasil : berdasarkan analisis singkat, tidak dilakukan analisis lebih jauh karena tidak diperlukan struktur data baru, karena sudah disediakan oleh OpenSteer versi terbaru

7. **Mengimplementasikan keseluruhan algoritma dan struktur data yang dirancang, dengan menggunakan framework OpenSteer**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

8. **Melakukan pengujian (dan eksperimen) yang melibatkan responde untuk menilai hasil simulasi secara kualitatif**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

9. **Menulis dokumen skripsi**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

10. **Mempelajari cara menggunakan fitur manipulasi obstacle yang disediakan oleh framework Opensteer versi terbaru**

Status : baru ditambahkan pada semester ini
Hasil : baru direncanakan karena framework Opensteer versi paling akhir baru selesai diinstall dan dilihat-lihat bagian contoh-contoh simulasinya

6 Pencapaian Rencana Kerja

Langkah-langkah kerja yang berhasil diselesaikan dalam Skripsi 1 ini adalah sebagai berikut:

1.
2.
3.

7 Kendala yang Dihadapi

Kendala - kendala yang dihadapi selama mengerjakan skripsi :

- Terlalu banyak melakukan prokratinasi
- Terlalu banyak godaan berupa hiburan (game, film, dll)
- Skripsi diambil bersamaan dengan kuliah ASD karena selama 5 semester pertama kuliah tersebut sangat dihindari dan tidak diambil, dan selama 4 semester terakhir kuliah tersebut selalu mendapat nilai E
- Mengalami kesulitan pada saat sudah mulai membuat program komputer karena selama ini selalu dibantu teman

Bandung, 22/11/2019

Muhammad Dipo Putra Wandara

Menyetujui,

Nama: Raymond Chandra Putra
Pembimbing Tunggal