Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

Team project ©2017
Dony Pratidana S. Hum | Bima Agus Setyawan S. IIP

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Dalam pengerjaan proyek piranti lunak, sistem atau aplikasi yang dibangun dapat mengalami perubahan yang bisa terjadi di tahapan mana saja dalam pengejaan proyek piranti lunak itu sendiri. Keinginan untuk merubah sistem tersebut akan terus bertahan sampai akhir siklus pengerjaan proyek piranti lunak terjadi (Bersof, dan kawan-kawan, 1980). Perubahan dapat terjadi disegala aspek mulai dari proses bisnis, permintaan teknikal (*technical requirements*), sampai ke model piranti lunak dan kode sumber (*source code*).

Proyek piranti lunak yang biasanya melibatkan tim yang terdiri dari dua orang atau lebih ini memerlukan sebuah mekanisme untuk mengontrol perubahan-perubahan yang terjadi di dalamnya. Hal tersebut bertujuan agar semua anggota tim yang terlibat dapat langsung melihat apa yang berubah dan mana yang merupakan hal yang harus dikerjakan pada fase berikutnya. Selain itu, mekanisme pengontrol perubahan tersebut juga bertujuan untuk menyatukan pekerjaan-pekerjaan yang dilakukan oleh beberapa orang pengembang (*developer*) agar didapatkan sebuah hasil piranti lunak yang dapat sedekat mungkin dengan *requirement* yang ada.

Mekanisme untuk mengatur perubahan yang terjadi terhadap kode sumber tersebut dengan *Software Configuration Management*. *Software Configuration Management (SCM)* atau dikenal juga dengan *Revision Control* merupakan proses mengidentifikasi konfigurasi sistem pada suatu waktu tertentu dengan tujuan mengendalikan perubahan konfigurasi dan menjaga integritas suatu konfigurasi

dalam sebuah siklus pengembangan sistem (Scott dan Nisse, 2001). Secara umum, kegunaan dari sebuah *SCM* adalah untuk mencatat perubahan yang terjadi terhadap piranti lunak yang dikembangkan agar dapat dilacak dan menghindari kesalahan yang dapat merusak keseluruhan tatanan piranti lunak itu sendiri. Tujuan lain dari penggunaan *SCM* adalah untuk membantu pengembang piranti lunak untuk mengatur kode sumber yang ditulis oleh pengembang itu sendiri, maupun pengembang lain.

Dalam sebuah *SCM*, terdapat beberapa mekanisme lain yang saling berkolaborasi menjadi satu kesatuan dan dapat membantu pengembang untuk mengembangkan piranti lunak yang ingin dibuat. Beberapa contoh mekanisme yang ada di dalam *SCM* adalah *version control* dan *data comparison*.

Data comparison merupakan alat yang digunakan untuk melihat perbedaan yang ada di antara dua buah data yang diberikan. *Data comparison* dapat digunakan untuk melihat perubahan yang terjadi antara dua buah versi piranti lunak yang dibuat oleh pengembangnya. *Data comparison tools* ini dianggap sangat berguna karena pengembang tidak perlu memindai satu persatu kode sumber mereka untuk mencari apa yang berubah pada kode sumber tersebut.

Penelitian ini telah menjadi fokus pada penelitian yang dilakukan oleh Adriansyah pada tahun 2007 dengan judul "PENERAPAN ALGORITMA X-DIFF DALAM MENDETEKSI PERUBAHAN PADA DOKUMEN XML." Namun, fokus pada penelitian tersebut adalah mendeteksi perubahan yang terjadi pada dokumen XML, sedangkan penelitian ini berfokus pada segala bentuk kode sumber dan tidak membedakan bahasa apa yang digunakan pada kode sumber tersebut. Selain itu, perbedaan lain dari penelitian tersebut dengan penelitian ini

adalah, algoritma yang digunakan. Penelitian tersebut menggunakan algoritma X-Diff, sedangkan penelitian ini menggunakan algoritma Myers Diff. Algoritma Myers Diff dipilih karena algoritma ini merupakan salah satu dari algoritma yang paling cocok digunakan untuk mengecek perubahan yang terjadi diantara dua buah data atau *file* yang memiliki dasar teks (*text based file*). Disamping itu, kecepatan dan kehandalan algoritma Myers Diff ini juga sudah diakui dengan masih digunakannya algoritma ini dalam beberapa *data comparison tools* merskipun sudah lama ditemukan.

Dengan tujuan membantu pengembang piranti lunak fokus terhadap pengembangan piranti lunak itu sendiri dan bukan berfokus pada pengontrolan kode sumber yang dimiliki, dilakukanlah penelitian untuk membangun *data comparison tools* dengan judul penelitian "Rancang Bangun *Data Comparison Tools* dengan menggunakan Algoritma Myers Diff untuk *Software Configuration Management*."

1.2 Rumusan Masalah

Penelitian ini berfokus untuk menjawab dua buah masalah yang telah dirumuskan yaitu.

1. Bagaimana merancang bangun *Data Comparison Tools* dengan menggunakan algoritma Myers Diff untuk *Software Configuration Management*?
2. Bagaimana menampilkan hasil perbedaan yang ditemukan antara dua buah data dengan baik dan mudah dilihat?

1.3 Batasan Masalah

Batasan masalah yang ditentukan untuk membangun aplikasi pada penelitian ini adalah sebagai berikut.

1. Data yang dibandingkan adalah data yang berbasis text atau binary, dan berfokus pada kode sumber (*source code*).
2. Aplikasi yang dibangun hanya membahas mengenai data comparison dan tidak membahas aspek *software configuration management* lain.
3. Penelitian hanya mencari perbedaan secara *syntax* menggunakan Myres diff algorithm dan bukan secara *semantic*.
4. Aplikasi yang dibangun hanya menampilkan perbedaan dari dua buah data yang diberikan dan bukan untuk menggabungkan (*merging*) dua buah data.

1.4 Tujuan Penelitian

Tujuan penelitian yang dilakukan ini adalah untuk menjawab permasalahan yang telah ditetapkan dengan uraian sebagai berikut.

1. Melakukan rancang bangun *Data Comparison Tools* dengan menggunakan algoritma Myers Diff untuk menunjang *Software Configuration Management*.
2. Menampilkan hasil perbedaan karakter yang ditemukan dari dua buah data dengan baik dan mudah dilihat dengan menggunakan uji penerimaan.

1.5 Manfaat Penelitian

Manfaat dari penelitian yang dilakukan adalah sebagai berikut.

1. Membantu pengembang proyek piranti lunak untuk mengontrol versi kode sumber yang dimilikinya agar dapat melacak dan mencatat perubahan yang terjadi terhadap kode sumbernya setiap waktu.
2. Membantu pengembang proyek piranti lunak untuk mengetahui perubahan yang dibuat oleh pengembang lain terhadap kode sumber yang ada.
3. Sebagai landasan bagi peneliti lain yang ingin meneliti topik yang serupa.

1.6 Sistematika Penulisan Laporan

Sistematika penulisan yang digunakan dalam penyajian laporan tugas akhir ini adalah sebagai berikut.

BAB I PENDAHULUAN

Bab ini berisi latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan laporan.

BAB II LANDASAN TEORI

Bab ini berisi teori-teori yang mendukung pembangunan aplikasi dalam penelitian ini, dan perancangan antar muka system.

BAB III METODE DAN PERANCANGAN SISTEM

Bab ini berisi medode penelitian, rancangan sistem *data flow diagram* dan *flowchart*.

BAB IV IMPLEMENTASI DAN UJI COBA

Bab ini berisi kebutuhan perangkat keras dan perangkat lunak untuk membangun dan menguji aplikasi, implementasi aplikasi, tampilan antar muka aplikasi, pengujian kondisi, pengujian validitas aplikasi, dan pengujian penerimaan aplikasi.

BAB V SIMPULAN DAN SARAN

Bab ini berisikan simpulan dari penelitian ini dan juga saran untuk pengembangan lebih lanjut aplikasi yang serupa.

UMN