BAB I
PENDAHULUAN

1.1 Latar Belakang

Kebutuhan manusia yang semakin kompleks menuntut berkembangnya teknologi komputer. Perkembangan ini membuat perangkat lunak komputasi harus bisa beradaptasi agar bisa memenuhi kebutuhan-kebutuhan tersebut. Semakin rumit kebutuhan yang ada, maka semakin rumit juga isi dari perangkat lunak yang dibuat. Ketahanan dari perangkat lunak menjadi salah satu faktor penting yang harus diperhatikan (Lehman & Ramil, 2002).

Coleman dkk (1994) menyatakan bahwa dalam industri pengembangan perangkat lunak, ketahanan *software* merupakan salah satu faktor yang penting untuk diperhatikan. Ketahanan perangkat lunak terhadap evolusi yang terjadi, melibatkan tingkat kompleksitas pengembangan suatu perangkat lunak. Jika perangkat lunak bisa bertahan terhadap kompleksitas kebutuhan yang diminta, maka perangkat lunak bisa dikatakan tidak mudah mati atau tergantikan.

Proses pengembangan perangkat lunak memerlukan pemeliharaan atau *maintenance*. Hal ini dikarenakan perangkat lunak yang dikembangkan berisiko memiliki kecacatan sehingga diperlukan perawatan agar perangkat lunak dapat terus berkembang. Menurut *IEEE Standard Glossary of Software Engineering Terminology* pemeliharaan perangkat lunak dapat dilakukan dengan memodifikasi sistem atau komponen dari perangkat lunak (IEEE, 1983). Tujuannya adalah untuk memperbaiki kesalahan yang telah ditemukan, meningkatkan performa dari perangkat lunak serta memungkinkan perangkat lunak untuk beradaptasi terhadap tuntutan perkembangan ke depannya. Perawatan perangkat lunak ini dilakukan berdasarkan permasalahan ataupun arah pengembangan dari perangkat lunak tersebut.

Perawatan pada perangkat lunak membutuhkan biaya berupa uang, tenaga dan juga waktu. Setiap perangkat lunak juga memiliki tingkat kesulitan yang berbeda-beda ketika melakukan perawatan. Tingkat kesulitan pada perawatan perangkat lunak ini bergantung kepada tingkat *maintainability* suatu perangkat lunak. Semakin mudah

pengembang melakukan perawatan pada perangkat lunak, maka perangkat lunak tersebut memiliki tingkat *maintainability* yang baik. Untuk mengukur tingkatan *maintainability* ini, sudah terdapat beberapa model pengukuran, misalnya sebuah model pengukuran yaitu *Maintainability Index*. Model *Maintainability Index* berusaha mengukur suatu perangkat lunak berdasarkan status dari *source code* perangkat lunak tersebut (Coleman dkk, 1994). Keluaran atau *output* dari model tersebut berupa angka nilai. Seperti beberapa model lainnya, *Maintainability Index* memiliki beberapa kekurangan yang terletak pada keluarannya yang kurang menggambarkan karakteristik dari *maintainability* sehingga pengembang dan pengguna kurang mengerti mengenai karakteristik *maintainability* yang terlibat dan cara untuk mengembangkannya (Heitlager dkk, 2007). Dikarenakan keterbatasan tersebut mulailah dikembangkan model lain yang lebih sesuai dalam menggambarkan karakteristik dari *maintainability* suatu *software*, misalnya *Software Improvement Group Maintainability Model*.

Peneliti menggunakan kode sumber sebagai objek pengamatan untuk menguji model ini. Objek pengamatan yang digunakan adalah kode sumber dari perangkat lunak *open source* JSONX-ORG/JAVA. Kode sumber dari perangkat lunak tersebut bisa diakses melalui laman GitHub JSONX-ORG/JAVA. Perangkat lunak JSONX-ORG/JAVA memiliki kode sumber dengan bahasa pemrograman utama Java yang sesuai untuk diuji menggunakan SIG *Maintainability Model* (Heitlager dkk, 2007). Melansir dari laman GitHub JSONX-ORG/JAVA, JSONX-ORG/JAVA merupakan *framework* (kerangka kerja) JSONX untuk Java yang menyediakan prosesor implementasi referensi, validator, dan *API runtime* untuk bahasa definisi JSON.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dibuat sebelumnya, dapat diambil rumusan masalah yaitu bagaimana cara mengukur *maintainability* dari JSONX-ORG/JAVA dengan SIG *Maintainability Model*? Bagaimana tren perubahan nilai *maintainability* pada keempat versi JSONX-ORG/JAVA dari hasil pengukuran menggunakan SIG *Maintainability Model*?

1.3 Tujuan dan Manfaat

Tujuan dari penelitian ini yaitu untuk mengukur dan membandingkan *maintainability* dari empat versi perangkat lunak JSONX-ORG/JAVA dengan menggunakan *SIG Maintainability Model*.

Manfaat yang didapatkan dari penelitian ini untuk adalah untuk menggambarkan proses dan konsistensi model pengukuran dari pengukuran *maintainability* perangkat lunak JSONX-ORG/JAVA berdasarkan nilai karakteristik kode sumber Versi 0.2.2, Versi 0.3.1, Versi 0.3.2, dan Versi 0.4.0 JSONX-ORG/JAVA menggunakan *SIG Maintainability Model* sehingga bisa menjadi referensi untuk penelitian keberlanjutan dalam mengukur tingkat *maintainability* dengan model SMM atau model lainnya.

1.4 Ruang Lingkup

Ruang Lingkup untuk skripsi ini adalah:

1. Objek pengamatan adalah *source code* perangkat lunak *open-source* JSONX-ORG/JAVA Versi 0.2.2, Versi 0.3.1, Versi 0.3.2, dan Versi 0.4.0 yang bisa diakses melalui laman GitHub.
2. Evaluasi kode sumber perangkat lunak dilakukan menggunakan *tools* PMD *Code Analyzer*, CLoC, Coverals, dan Simian

1.5 Sistematika Penulisan

Sistematika Penulisan berisi gambaran umum dari penulisan skripsi, sistematika penulisan dapat dijabarkan sebagai berikut:

BAB I PENDAHULUAN

Bab I berisikan latar belakang masalah, rumusan masalah, tujuan dan manfaat, ruang lingkup, serta sistematika penulisan dari skripsi ini.

BAB II LANDASAN TEORI

Bab II berisi landasan teori yang digunakan dalam proses penyusunan skripsi ini. Teori yang digunakan untuk bahasan mencakup tinjauan pustaka mengenai *Software Maintainability* dan model pengukuran *Software Maintainability*. Model pengukuran yang digunakan adalah SIG *Maintainability Model* (SMM). Model SMM mengukur nilai *maintainability* berupa *Analysability*, *Changeability*, *Stability*, dan *Testability* berdasarkan *source code characteristic* (*properties*) dari perangkat lunak. Objek penelitian yang digunakan adalah perangkat lunak *open source* JSONX-ORG/JAVA dan *tools* yang digunakan untuk menganalisis kode sumber adalah Coverals, Simian, PMD *Code Analyzer*, CLoC, Excel, Visual Studio Code.

BAB III METODOLOGI PENELITIAN

Bab III berisikan metodologi penelitian yang dilakukan dalam melaksanakan penelitian ini. Bagian ini berisi gambaran umum dan alur pengerjaan secara keseluruhan dari penelitian yang dilakukan. Langkah-langkah yang dilakukan adalah pengumpulan objek penelitian, langkah-langkah dalam menganalisis kode sumber menggunakan *tools*, mengukur karakteristik kode sumber menggunakan SIG *Maintainability* Model, serta menganalisis hasilnya.

BAB IV HASIL DAN PEMBAHASAN

Bab IV berisikan pembahasan hasil penemuan yang didapatkan selama penelitian. Bab ini berisikan implementasi langkah-langkah yang sudah dijabarkan pada Bab Metodologi Penelitian.

BAB V PENUTUP

Bab V berisikan penutup yang terdiri dari kesimpulan, saran, dan penelitian lanjutan yang dapat dilakukan.