BAB II
TINJAUAN PUSTAKA

Pada bab ini dipaparkan sejumlah konsep yang digunakan dilakukan dalam penyusunan tugas akhir ini. Konsep-konsep yang digunakan dalam penelitian ini terkait dengan FOSS, manajemen proyek, proses perangkat lunak, Configuration Management, *fork* dan turunan, distribusi GNU/Linux, serta OSS 2.0.

2.1. FOSS

FOSS (*Free/Open Source Software*) adalah sebuah istilah yang biasa digunakan dalam paper-paper ilmiah yang membahas tentang fenomena Perangkat lunak Bebas (*Free Software*) dan Perangkat Lunak Kode Sumber Terbuka (*Open Source Software*). Terlepas dari perdebatan mengenai perbedaan definisi antara *Free Software* (FS) dan *Open Source Software* (OSS), dalam tulisan ini kedua istilah tersebut dinyatakan dengan FOSS.

Perangkat lunak *open source* tidak hanya menjabarkan bahwa kode-sumbernya terbuka bagi siapa saja, melainkan juga harus mengikuti sekumpulan kriteria-kriteria berikut [OSI09]:

1. *Free Redistribution*

   Artinya, lisensi perangkat lunak tidak membatasi pihak manapun untuk menjual atau memberikan secara cuma-cuma atas perangkat lunak yang bersangkutan kepada pihak lain.

2. *Source Code*

   Program yang bersangkutan harus disertakan dengan kode sumber, dan lisensi harus memungkinkan distribusi dalam bentuk kode sumber maupun program hasil kompilasi.

3. *Derived Works*

   Lisensi harus mengijinkan pengubahan terhadap perangkat lunak maupun produk hasil turunan, dan memperbolehkan produk-produk hasil modifikasi ataupun turunan tersebut dengan lisensi yang sama dengan lisensi perangkat lunak yang asli.

4. *Integrity of the Author's Source Code*

   Lisensi boleh memperketat kode sumber untuk tidak didistribusikan dalam bentuk yang telah diubah jika lisensi tersebut mengijinkan distribusi berkas-berkas tambalan berbentuk kode sumber yang bertujuan untuk memodifikasi program pada tahap pembangunan.

5. *No Discrimination Against Persons or Groups*

   Lisensi tidak boleh mendiskriminasikan siapapun atau kelompok manapun.

6. *No Discrimination Against Fields or Endeavor*

   Lisensi tidak boleh membatasi tujuan dari pemanfaatan perangkat lunak terkait

7. *Distribution of License*

   Hak-hak yang dijelaskan di dalam lisensi harus dapat berlaku bagi semua pihak yang menerima program hasil redistribusi tanpa perlu lisensi-lisensi tambahan

8. *License Must Not Be Specific to a Product*

   Lisensi yang melekat pada program tidak boleh tergantung kepada lisensi program sebagai bagian dari distribusi perangkat lunak.

9. *License Must Not Restrict Other Software*

   Lisensi tidak boleh membatasi perangkat lunak lain untuk dapat disertakan bersama perangkat lunak yang diberi lisensi

10. *License Must Be Technology-Neutral*

Lisensi tidak boleh mensyaratkan teknologi tertentu maupun antarmuka khusus.

2.2. *Manajemen Proyek*

Salah satu cara memandang pengembangan FOSS adalah melalui sudut pandang ilmu manajemen proyek. Hal ini dapat dilakukan karena secara umum pengembangan FOSS memiliki bentuk berupa proyek pengembangan perangkat lunak. Meskipun berbagai aspek manajemen seperti sumberdaya manusia, waktu, dana, dan lain sebagainya bisa jadi tidak sesuai dengan manajemen proyek pada pengembangan perangkat lunak konvensional.

Manajemen Proyek didefinisikan sebagai pengaplikasian pengetahuan, kemampuan, dan teknik pada aktivitas-aktivitas proyek untuk memenuhi kebutuhan-kebutuhan proyek. Manajemen proyek dilakukan melalui aplikasi dan integrasi dari proses-proses manajemen dalam fase inisiasi, perencanaan, pelaksanaan, pengawasan, hingga penutupan proyek [PMI04]. Namun, definisi tersebut masih terlalu generik jika hendak digunakan sebagai kacamata untuk memandang proyek perangkat lunak bebas.

Lebih spesifik lagi, Fogel [FOG07] memaparkan sejumlah isu penting yang mesti diperhatikan dalam proyek perangkat lunak bebas, yaitu:

- Infrastruktur Teknis

  Infrastruktur Teknis adalah perangkat-perangkat teknis yang diperlukan untuk melakukan pengembangan perangkat lunak bebas. Perangkat-perangkat ini dapat berupa perangkat lunak lainnya maupun infrastruktur perangkat keras.

- Infrastruktur Sosial dan Politik

Infrastruktur Sosial dan Politik adalah kondisi-kondisi sosial yang mendukung berjalannya suatu proyek FOSS. Faktor yang menjadi infrastruktur sosial politik bisa berupa dukungan sponsor, mekanisme penerimaan kontributor baru, hirarki proyek tersebut, dan mekanisme mengatasi konflik.

- Dana

  Sebuah proyek perangkat lunak yang bertahan harus memiliki sumber dana untuk mendukung kegiatan proyek tersebut, tak terkecuali proyek FOSS. Pendanaan ini dapat mempengaruhi arah kebijakan sebuah proyek FOSS dan kelangsungannya.

- Komunikasi

  Kolaborasi yang terjadi dalam pengembangan proyek FOSS melibatkan banyak pihak yang secara geografis berpisah. Penataan komunikasi yang baik mendukung proses pengembangan dalam. Setiap galat yang ada dapat ditangani dengan cepat dan proses pengambilan keputusan menentukan arah kebijakan proyek FOSS. Tanpa adanya komunikasi antara pihak-pihak yang terkait maka proses pengembangan perangkat lunak tidak akan berjalan lancar. Bahkan, akibat kesalahan komunikasi proyek FOSS mengalami kegagalan dan terhenti.

- Pemaketan, Perilisan, dan Pengembangan Harian

  Proses pemaketan, perilisan, dan pengembangan harian dari proyek FOSS merupakan siklus pengembangan perangkat lunak yang terjadi secara terus menerus. Manajemen dari siklus ini menjamin ketersediaan produk hasil dari proyek FOSS secara periodik.

- Manajemen Kontributor

Sifat utama dari proyek FOSS adalah saling berbagi dan setiap orang bisa secara sukarela berkontribusi terhadap proyek tersebut. Berkembangnya sebuah proyek FOSS ditandai dengan semakin banyaknya individu maupun organisasai yang ikut serta di dalamnya. Agar tidak terjadi tumpang tindih dan gesekan sosial, sebuah proyek FOSS memiliki pengaturan atau semacam mekanisme untuk setiap orang dapat berkontribusi secara positif.

- Lisensi, Hak Cipta, dan Paten

  Pengembangan proyek FOSS juga berkaitan dengan aspek legal seperti lisensi, hak cipta, dan paten. Lisensi dari sebuah produk yang menggunakan proyek FOSS dengan lisensi yang berbeda akan mempengaruhi kontributor yang terkait pada pengembangan produk tersebut. Penggunaan lisensi bebas dapat membatasi kontribusi produk yang akan dikomersilkan, begitu pula sebaliknya, penggunaan lisensi terbatas akan membuat kontributor dengan ideologi tertentu tidak ingin berkontribusi terhadap proyek FOSS tersebut. Pengembangan proyek FOSS juga dapat bermasalah jika hendak menggunakan teknologi yang sudah dipatenkan terlebih dahulu.

2.3. Proses Perangkat Lunak

Proyek FOSS, yang juga termasuk pengembangan perangkat lunak, tentunya menerapkan juga proses pengembangan perangkat lunak. Untuk memahami proses yang dilakukan dalam sebuah pengembangan perangkat lunak, sudut pandang yang dapat digunakan adalah sudut pandang dari ilmu Rekayasa Perangkat Lunak.

Rekayasa Perangkat Lunak adalah sebuah disiplin ilmu yang menyediakan kerangka kerja yang dapat diikuti untuk membangun perangkat lunak dengan kualitas tinggi. Untuk itu, sejumlah permodelan proses pengembangan perangkat

lunak telah dibakukan. Model-model tersebut antara lain adalah sebagai berikut [PRE05]:

- *Waterfall Model*
- *Incremental Model*
- *RAD Model*
- *Prototyping*
- *Spiral Model*
- *Concurrent Development Model*
- *Component-Based Development*
- *Formal Method Model*
- Aspect Oriented Software Development

2.4. Configuration Management

Pengembangan proyek FOSS yang melibatkan kolaborasi banyak orang menuntut adanya pencatatan mengenai perubahan-perubahan apa yang telah dilakukan terhadap produk perantara maupun produk akhir. Oleh karena itu peranan *Configuration Management*, terutama *Source Code Versioning System* merupakan sesuatu yang dibutukan di dalam proyek pengembangan FOSS.

Configuration Management adalah identifikasi unik, penyimpanan terkontrol, kontrol perubahan, pelaporan status dari produk kerja perantara pilihan, komponen-komponen produk, serta produk-produk selama daur hidup sebuah sistem [HAS02]. Salah satu contoh dari *Configuration Management* yang dibahas di dalam penelitian ini adalah *Source Code Versioning System*.

Source Code Versioning System/Version Control System adalah kombinasi dari sejumlah teknologi dan praktek dalam melakukan pelacakan dan mengatur perubahan terhadap artifak-artifak proyek seperti berkas kode sumber, dokumentasi, maupun halaman web [FOG07].

Ada dua model repositori yang diterapkan untuk kebutuhan *Version Control*

System (VCS), yakni model tersentralisasi dan model desentralisasi. Model tersentralisasi mengharuskan hasil kerja dan perubahan terhadapnya untuk diletakkan di dalam sebuah server yang dapat diakses oleh komputer-komputer yang digunakan untuk pengembangan melalui jaringan.

Sebaliknya, model ter-desentralisasi melakukan pencatatan pada sisi komputer yang digunakan oleh pengembang (klien). Salah satu contoh dari aplikasi VCS yang menerapkan metode terdesentralisasi semacam ini adalah Bazaar VCS.

2.5. Ohloh

Ohloh adalah sebuah organisasi yang didirikan oleh Jason Allen dan Scott Collison pada tahun 2006. Tujuan dari didirikannya Ohloh antara lain adalah menyediakan informasi visibilitas pada pengembangan perangkat lunak, khususnya FOSS. Ohloh memiliki sebuah situs *web* (http://ohloh.net/) yang berisi informasi tentang aktivitas kerja dalam pengembangan sejumlah proyek FOSS (Ohloh mencatat lebih dari 300 ribu buah proyek pada Juni 2009). Informasi yang ditampilkan berasal dari penghimpunan dan analisa data yang diperoleh dari *Version Control* yang digunakan dalam pengembangan proyek FOSS yang terdaftar.

Dalam penyusunan tugas akhir ini, penyusun melakukan analisa terhadap catatan dalam *Version Control* seperti yang dilakukan oleh Ohloh. Namun karena Ohloh belum memberikan dukungan untuk Version Control yang digunakan oleh kelompok kerja pengembangan FOSS yang menjadi subjek dari penelitian (yaitu *Bazaar VCS*), maka penghitungan data dilakukan secara manual.

2.6. Fork dan Turunan

Proyek FOSS yang menjadi subjek dari penelitian ini adalah proyek FOSS yang

didasari pada hasil dari proyek FOSS yang lain. Ada dua cara dikenal dalam pengembangan FOSS dari FOSS yang lain, yaitu dengan cara membuat *fork* dan turunan.

Code Forkability adalah istilah di dalam FOSS yang bermakna kemungkinan bagi siapa saja untuk memperoleh kode sumber dari suatu proyek FOSS dan menggunakannya untuk proyek FOSS yang lain [FOG07]. Sebuah proyek FOSS disebut sebuah *fork* apabila pembuatnya bermaksud menggantikan atau menandingi proyek FOSS aslinya [WHE07].

Derivative work (karya turunan) adalah sebuah karya yang didasarkan pada sebuah karya atau beberapa karya yang sudah ada, misalnya sebuah terjemahan, aransemen musikal, dramatisasi, fiksionalisasi, versi film, rekaman suara, reproduksi seni, abridgement, condensation, atau dalam bentuk-bentuk yang lain yang dapat disiarkan ulang, diubah, atau diadaptasikan. Sebuah karya yang berisi revisi editorial, anotasi, elaborasi, atau modifikasi lainnya, yang secara keseluruhan merepresentasikan sebuah karya hasil gubahan, termasuk ke dalam *derivative work* [USC07].

2.7. Distribusi GNU/Linux

Hasil dari proyek FOSS yang dibahas dalam penelitian ini berupa sebuah perangkat lunak yang disebut dengan distribusi GNU/Linux. Bagian ini berisi penjelasan mengenai apa itu distribusi GNU/Linux, macam-macam distribusi GNU/Linux yang telah ada, serta persamaan dan perbedaan antara distribusi satu dengan distribusi yang lainnya.

Distribusi GNU/Linux adalah sistem operasi berbasiskan perangkat lunak bebas dari proyek GNU (http://www.gnu.org/), kernel sistem operasi Linux, dan sejumlah perangkat lunak bebas lainnya yang telah melalui serangkaian proses kompilasi dan konfigurasi [MCC99]. Contoh dari distribusi GNU/Linux yang populer saat ini adalah Debian GNU/Linux, Fedora, Ubuntu Linux, SUSE,

Knoppix, Slackware, dan lain sebagainya.

Pada dasarnya sebagian besar distribusi linux memiliki komponen inti berupa kernel sistem operasi yang sama, yakni kernel Linux. Selain itu beberapa aplikasi yang dimasukkan ke dalam distribusi-distribusi yang berbeda sebenarnya juga berasal dari sumber yang sama. Misalnya saja *Apache Web Server*, *Postfix Mail Transfer Agent*, *GNU Compiler Collection*, dan lain sebagainya.

Perbedaan yang menonjol dari tiap distribusi-distribusi besar biasanya terletak pada Sistem Manajemen Paket (dijelaskan pada bagian berikutnya) yang digunakan, *User Interface*, *Desktop Manager*, jumlah aplikasi yang di-bundle dalam media distribusi, dan seterusnya.

2.8. OSS 2.0

Pandangan mengenai FOSS sebagai sebuah gerakan sukarela dari sekelompok hacker yang bekerjasama untuk menghasilkan perangkat lunak yang berkualitas sudah tidak relevan lagi. Hal tersebut disebabkan karena ada indikasi bahwa pada masa mendatang fenomena open source akan bergeser menuju bentuk yang lebih mainstream dan komersial, yang disebut sebagai OSS 2.0 [FTZ06].

Pergeseran paradigma tersebut terlihat dari indikasi yang terdapat pada faktor proses yang terlibat dan faktor produk yang dihasilkan oleh gerakan open source sejauh ini. Secara singkat, Fitzgerald mengemukakan pergeseran yang terjadi seperti yang digambarkan dalam tabel 2.1 dan tabel 2.2.

Tabel 2.1: Perbedaan antara F/OSS dengan OSS 2.0 dari segi proses

PROSES	F/OSS	OSS 2.0
Development Lifecycle	● Fase *Planning*. Tidak ada strategi yang mengarah perolehan keuntungan melalui kompetisi. ● Fase *Analysis*. Biasanya merupakan bagian dari pengetahuan-pengetahuan konvensional seputar pengembangan perangkat lunak. ● Fase *Design*. Berbasis pada prinsip-prinsip modularitas untuk mengakomodir beragamnya pendapat. ● Fase *Implementation*. Terdiri dari beberapa sub-fase: ○ *Code* ○ *Review* ○ *Pre-commit test* ○ *Development release* ○ *Paralel debugging* ○ *Production release*	● Fase *Planning*. Terdapat susunan perencanaan strategis oleh perusahaan untuk berkompetisi dalam pasar ● Fase *Analysis* dan *Design*. Lebih kompleks karena mengarah ke domain yang 'vertikal', yang mana seringkali (business) requirements bukan merupakan perkara yang dapat dimengerti secara menyeluruh. ● Fase *Implementation* memiliki sub-fase yang mirip dengan F/OSS. Tetapi secara keseluruhan, pengembangannya tidak terbuka untuk umum ● Para pengembang dibayar untuk mengerjakan *open source*

Tabel 2.2: Perbedaan antara F/OSS dengan OSS 2.0 dari segi Produk

PRODUK	F/OSS	OSS 2.0
Domain Produk	Infrastruktur horizontal (sistem operasi, *utilities*, *compiler*, DBMS, *web server*, *print server*)	Aplikasi berbentuk Sistem Informasi dalam domain vertikal
Strategi Bisnis Utama	● *value added service enabling* Menyediakan layanan tambahan (berbayar) terhadap produk. ● *Loss leader/market creating* Membuka pasar yang baru dengan produknya.	● *value added service enabling, bootstrapping* ● Membuka pasar: *loss leader*, *dual product/licensing*, *cost reduction*, *accessorizing* ● Membentuk komunitas pengembang ● Mengangkat *brand open source*
Layanan Produk	Tidak jelas. Kebanyakan bergantung pada *mailing list/bulletin board*, selain itu ada juga perusahaan yang memberikan layanan bantuan	Pelanggan membayar untuk layanan yang berbentuk '*whole-product*'
Lisensi	● GPL, LGPL, Artistic License, BSD, MPL ● Hubungan antar lisensi yang menular/*viral*	● Sangat bervariasi ● Hubungan antar lisensi yang bersifat timbal balik ('Resiprok')