2.2 FreeBSD プロジェクトに見るオープンソース開発プロジェクトの実際

大阪大学大学院情報科学研究科　松下誠

本稿では、オープンソースソフトウェア開発の中でも比較的歴史が長く、開発形態も成熟している FreeBSD の開発を例に挙げて、その組織やプロセス等について順を追って見ることとする。著者はこれまで FreeBSD に関するさまざまな活動に携わってきており、以下はそれらを通じて得られた、数多くの経験に基づいて書かれている。

1. FreeBSD とは

FreeBSD は、1993 年に最初のリリース(1.0)が行われた、いわゆる BSD UNIX 系オペレーティングシステムのひとつであり、Intel x86 や Alpha PC 上で動作する。UCB (University of California, Berkeley) の CSRG (Computer Software Research Group) によって配布されていた 4.3BSD NET/2 リリースを元に、OS として不足していた部分を補う形で作成された 386BSD をその源流として持つ。

Linux の世界ではカーネルを含めた各種コンポーネントがそれぞれ独立に開発され、ディストリビュータがそれらをまとめて OS としての体裁を整えているが、FreeBSD などの BSD UNIX では、基本的な部分（カーネル、コンパイラ、基本ツール等）はすべて FreeBSD のソースコード中に含まれている。このため、FreeBSD 自身のソースコード規模はおよそ 250MB と、単一のオープンソースソフトウェアとしては比較的規模が大きい。BSD UNIX は、その昔より TCP/IP の実装に優れているとされ、サーバ系などを中心に広く用いられている。たとえば、Yahoo! や Apache Project の Web サーバとして FreeBSD が用いられていることが良く知られている。

2. 開発組織

一口に「開発組織」と言っても、オープンソース開発プロジェクトの場合「どこからどこまで」を開発組織として考えても良いのか、という問題が存在する。これは、世界中にちらばっている作業者が自由に参加、あるいは離脱できるということや、雇用契約などの明示的な契約を結んでいないため、明確な線引きを行うことが難しいことに由来する。

本稿では、FreeBSD のプロダクトを直接修正できる開発者を「FreeBSD プロジェクトの開発者」として考え、これをもって開発組織が構成されるという考え方に立つ。この定義の場合、いわゆる「パッチを書いて送った」程度の開発者は含まれないことになる。しかし、開発組織という視点から見る場合、開発者が明示的に組織の一員であることを自覚しているかどうかが重要であると考えられること、また、他の類似したオープンソース開発でも似たような定義が用いられていることから、この定義を用いるものとする。

以下、組織階層のより低位から順番に開発組織を見ていくこととする。

開発者（コミッタ、committer）

FreeBSD プロジェクトには、本稿執筆時点でおよそ 300 人程度の「コミッタ(committer)」と呼ばれ

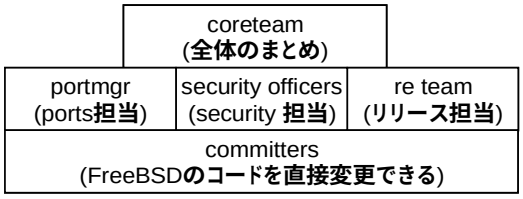

図 1: FreeBSD の開発組織

る開発者が存在している。コミッタは、FreeBSD のプロダクトを直接修正する権限がある。300 人という規模は、一般のソフトウェア開発から考えると比較的規模が大きいと言えるが、コミッタはそれぞれ自分が主に作業する領域があるということや、コミッタは常に FreeBSD の作業を行っているわけではない（ほとんどのコミッタは自分の余暇時間を利用して作業を行っている）ため、この規模でも人数が不足しているという感じである。

FreeBSD プロジェクトに多くの貢献（特に、修正内容の提供など）をしている人の中から、既存のコミッタが推薦することによって新たなコミッタが誕生する。新たなコミッタの申請は基本的にコアチーム（後述）に提出され、コアチームがそれを承認した時点でコミッタへ就任することが認められるが、ports[1]関連の作業をするコミッタは、これとは別に portmgr と呼ばれる ports 関連の責任者グループがコアチームの代わりに承認することとなっている。このほかにも、セキュリティ周りの問題を担当する security officers や、FreeBSD のリリース作業全体を管理する re team (release engineering team)など、作業に応じていくつかの役割がある。なお、一人の作業者が複数の担当となることも少なくない。

現在、コミッタのおよそ 6 分の 1（約 50 人）が日本人であり、アメリカを除くと比較的多くのコミッタが日本から輩出されている。

コアチーム (core-team) と管理グループ

コアチームは FreeBSD プロジェクト全体の方向を定め、プロジェクト自体を成功へと導くための舵とり役を果たす組織であり、原則 9 人で構成されるコアチームはコミッタ同士の互選によって選ばれ、2 年の任期が定められている。

また、コアチーム以外にも、各作業分野ごとにその分野での責任者が適宜定義されている。主なものとして portmgr, security officers, release engineering team がある。

FreeBSD プロジェクト全体を統括するコアチームと同様な役割を持つが、FreeBSD ports にその範囲を限った組織として portmgr(port manager)が存在している。従来はコアチームがその役割を担っていたが、FreeBSD ports 自体があまりにも巨大となってしまい、その管理コストが膨大なものとなってきたことと、ports 自体は FreeBSD 配布物の一部でありながら、ports によって管理されるアプリケーションは FreeBSD 以外の組織によって開発されたものであり、アプリケーションの性格が異なることから、別の組織として portmgr という役割を作り、コアチームは portmgr へ権限を委譲する、という形をとっている。FreeBSD は OS という性格上セキュリティ問題にも注意を払っており、情報収集や実際に問題が発生した場合の対応などは security officers が他に優先してさまざまな指示を行っている。さらに、定期的なリリースについては、利用者が実際に用いるバイナリ作成を責任を持って行い、リリース前の品質管理作業やドキュメント整備などを担当する release engineering team が存在する。

この他、各アプリケーションに依存した開発プロジェクト、マニュアルやハンドブックの執筆を主に行っているドキュメンテーションプロジェクトなど、各開発の側面に応じてさまざまな組織があり、それぞれが相互に連携しながら開発作業を行っている。プロジェクト内、あるいはコミッタ同士で何らかの論争が起き、収拾がつかなくなった場合には、最終的な判断はコアチームに委ねられる。

3. 開発プロセス

一般的なソフトウェア開発組織とは異なり、組織があまり明確に定義されないことや、組織全体を厳格に統括する組織がないこと、また、開発者の能力や技術背景に大きな隔たりがあることなどから、FreeBSD プロジェクト全体の開発プロセスを、明確に定義することは非常に困難である。しかし、小さな、あるいはおおまかなプロセスに分けるこ

[1] Apache など、FreeBSD 自体には含まれていない各種アプリケーションについて、入手、コンパイル、インストールなどの方法を記述したもの。利用者は ports に登録されているアプリケーションであれば、make コマンドで容易にインストールすることができる。

とは可能である。以下では、規模に応じてどのような開発プロセスが存在しているかを述べる。

全体プロセス

前述した通り、コミッタ全体で何らかの話し合いを行い、スケジュール表を作成して開発を行うということは現実的ではなく、実際にそのようなことは行われていない。プロジェクト全体で作業内容について合意が取れている内容としては、Committers' Guide と呼ばれる文書に書かれているものがある。この文書は、FreeBSD のプロダクトを修正するための手順といった非常に具体的な内容から、開発者としての心構え（他の開発者を尊重しましょう、など）、プロジェクト内で活動する際のルールといった非常に抽象的な内容が記述されている。

唯一、FreeBSD プロジェクト全体で定められているプロセスをあげるとすれば、それはリリースにかかわるプロセスである。開発の結果として FreeBSD のリリース版を作成する際には、あらかじめリリース日を確定させた上で、そのリリース日までに間に合うように全体の作業が進められる。リリース自体は数ヶ月単位で定期的に行われているため、この全体作業は安定した間隔で定期的に行われることとなり、これが FreeBSD 全体を安定して進化させることができる要因となっているのではないかと考えられる。リリース作成時のプロセスについては、(3)リリース管理で別途述べる。

チーム･個人プロセス

全体プロセスがあまり明確に定義されないことから、FreeBSD プロジェクトにおける開発作業の多くは個人単位、あるいは、お互いに理解しあっている複数のコミッタ等によるチーム単位で開発が行われることとなる。

作業の種類としては、大まかに「デザイン」、「コーディング」、「テスト」などといった、ウォーターフォールモデルの各フェイズに相当する作業があり、これらの作業を行う際には、電子メール等を用いたコミュニケーション、実際に動作する、あるいは動作する可能性があるソースコードを相互にやり取りすることによって、どういう実装にするべきか、あるいは、実際に動作しているかの確認作業が進められる。ある問題に対して、技術的に競合する複数の案が出されることも少なくはないが、この場合にはどちらの案が優れているかを技術的な観点から議論し、お互いに納得した上でどれかの案を選択し、それにしたがって実装を行うこととなる。

また、各アプリケーション、あるいはその一部にはメンテナ(maintainer)と呼ばれる「主にこのソースコードを管理する人」が定められていることがある。この場合、一般的にはコミッタがメンテナに対して確認を求め、その承認が得られた時点で実際の修正を行うといった、緩やかな管理体制がとられている。

リリース管理

定期的に行われる FreeBSD のリリース作業は、その作業を専門に行うリリースチーム(re：release engineering)によって現在行われている。リリースチーム内では、全体管理と各アーキテクチャ用のリリース版作成、またパッケージ作成作業といった複数の役割にが分担されている。実際には一人の作業者が複数の役割を果たしているため、例えば 4.5-RELEASE の場合、6 人でこのリリースチームが構成されていた。

リリースチームは、コアチームが定めたリリース日を守るために全体の作業を管理することとなる。まず、コアチームがかなり早い時点、通常数ヶ月以上前にリリース日程を定め、これを広く広報する。本稿執筆時点では 4.8-RELEASE のリリース時期（2003 年 2 月）まですでに公となっている。このリリース時期の早期決定については、FreeBSD プロジェクトに特徴的なものであると思われる。以下は、おおまかな FreeBSD リリーススケジュールの流れである。

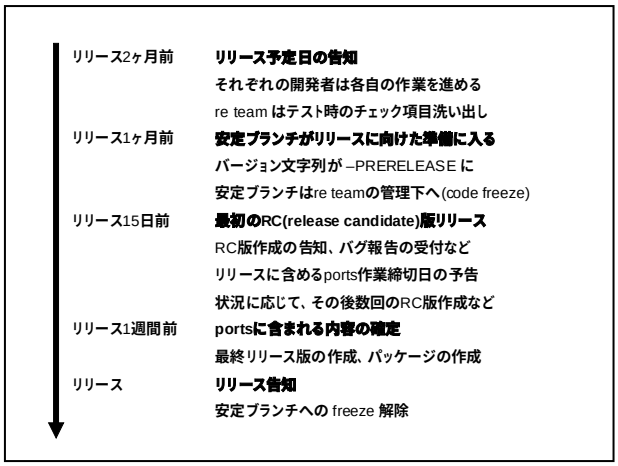

図 2: リリーススケジュールの例

リリース日が迫ってきた時点で、リリースチームが本格的な作業を開始する。まず、1 ヶ月前にその時点のソースコードを使ってプレリリース版を作成する。この時点で、次のリリースに採用する機能等が確定され、以降リリース版に向けての修正はすべてリリースチームの承認が必要となる。このような状態はコードフリーズ(code freeze)と呼ばれている。

コードフリーズ中は、既存のソースコードに存在しているバグを取り除く作業を開発者が集中して行うこととなる。プレリリース版によって発見されたバグは、直ちに対応されることが望まれている。ある程度の修正が行われた時点、一般的にはコードフリーズされてから半月後に、最初の RC 版の作成が行われる。RC 版はプレリリース版よりもさらに多くのユーザによって使われるため、より詳細な問題を発見するために役立てられる。以降、数回にわたって RC 版の作成を行い、目だった問題が発見されなくなった時点で、最終版であるリリース版が作成される。

リリース版作成の際には、FreeBSD 自身だけではなく、ports より作成されるアプリケーションパッケージの作業や、ハンドブックなどのドキュメント関連の作業が並行して行われる。リリース版が確定すると、ftp 配布のマスターサイトにそれが置かれ、これが世界中のミラーに広まった時点でリリース告知が行われる。

「リリース日を守る」という時間制約を満たすためには、いろいろ努力が払われている。まず、前述したコードフリーズ中のリリースチームによるレビューである。これにより、システムにとって重要な修正は基本的に禁止され、細かいバグや挙動に変更を加えないことが保証されている内容のみがソースコードに加えられることとなる。もちろん、どうしても間に合わない、という場合にはリリース時期が延期されることになるが、過去の経験では、長くても 1 ヶ月程度の延期でとどまっている。

もう一つの努力として、定期的なリリース版の提供があげられる。これは、毎日最新の FreeBSD ソースコードを用いて、実際のリリース版と同じ形態の配布物を自動的に作成して提供するものである。これを用いることにより、インストールの際にのみ発覚する問題を、コードフリーズ中以外でも容易に発見することが可能となる。また、すでに FreeBSD が稼動している PC では、最新のソースコードを入手して新しい FreeBSD を生成することが可能であるため、最新のソースコードにのみ存在する問題を未然に発見するのに役立っている。

4. 開発ツール

FreeBSD の開発では、他のオープンソース開発と同種のツールが用いられている。まず、FreeBSD のプロダクトは CVS (Concurrent Versions System)[1]によって管理されている。前述したコミッタは、CVS によって管理されているプロダクトを専用のツールを用いて修正する。CVS の利用状況については後述する。

コミッタ相互はもちろん、一般的に議論や情報交換を行う場合には、電子メールが広く使われている。電子メールを特定多数に配布するためのメーリングリストはもちろん、過去の議論を追いかけるためのメールアーカイブが整備されている。また、FreeBSD のリリース告知やセキュリティ情報の告知などにも電子メールが用いられている。開発者相互のコミュニケーションには IRC(Internet Relay Chat)も広く用いられている。

その他の一般的な情報提供については、主に WWW が用いられている。メールアーカイブも WWWによって閲覧することができるほか、CVS によって管理されているプロダクトも WWW 越しに閲覧することが出来るため、最新のソースコードはもちろん、過去の開発経緯についても生の情報がWWWによって提供されている。

本稿が他のオープンソース開発の状況を調査する際の参考になれば幸いです。

参考文献

[1] Concurrent Versions System, http://www.cvshome.org/