index

オープンソースソフトウェアの育て方感想

書籍「オープンソースソフトウェアの育て方」を読んだので感想を書きます。
http://producingoss.com/ja/

書籍の内容のほとんどは、プロジェクトが失敗しないようにするための、するべからず集になっている感じです。

オープンソースのプロジェクトが失敗に終わる確率は90–95%くらいで、その失敗のほとんどの原因はプロジェクトを続ける人がいなくなってしまう、プロジェクトは存在しているが活動はしていない、ゾンビ状態になる原因がほとんどだそうです。

このことから、オープンソースプロジェクトが成功するためには、今いる開発者はできるだけ留まってもらうようにし、また新参者はすぐ開発に参加できるような環境をつくることが大前提のようです。

特に印象に残っているのが、メーリングリストにリンクを張りましょうという話。最初は何を言ってるのかわかりませんでしたし、そんなに重要なことなのか？と思いましたが、メーリングリストは大勢の人が見るため、リンクを張らないことによって、読んだ人がリンク先の内容を調べる手間が一人ひとりにかかるため、その結果リンクをしなかったという手間を惜しんだだけで、１人月に匹敵するコストがかかることがある。という話です。

それはそうです。資源は限られているのですからいちいちそんなことをしてたら、当然終わるものも終わらず、人は居なくなりますよね。

書籍「オープンソースソフトウェアの育て方」はそういったプロジェクトが失敗しないようにあれこれ気をつけましょう、ということが書いてある書籍でした。

以下は知らなかった、参考になったメモ

- ドキュメントが必要な理由は新参者がプロジェクトに参加しやすくするため。
- 新参者がコミュニティのウェブサイトを訪問して30秒で参加するかどうかを決めるため、最初の声明文は重要になる。
- プロジェクトを世に出すにあたって最初に見てもらうところがウェブサイトであり、その印象がプロジェクトにも持ち込まれるため、見た目もコード以上にプロジェクトを成功させる重要な要因の一つになる。
- ハードウェアアーキテクチャの差がなくなり、ソフトウェアで差別化を図って売っていかなければ生き残れなくなったメーカーは、ソフトウェアの改変を規制し始めた。それがオープンソースの始まり。
- ガイドラインを作成するだけではダメで、それを必要としている人たち、プロジェクトに関わりのない人たちも含めて目に付きやすいところに掲げ、明確にわかるようにしなければならない。
- 新参者にライセンスについて探させない、マウスクリックさせない、負担をかけないように、予めライセンスはソースとREADMEに書いておく。
- 本来一匹狼である開発者たちを結びつける唯一の原動力は、個別にやるよりも協調したほうが、よりよいものができるという共通認識のみ。
- 開発者向けには開発者向けのガイドラインを用意する

- 重要な決定を内輪でこっそり行うのは、まるでそのプロジェクトに「貢献者よけスプレー」を振りまくようなもの、特別に隠す必要があるもの以外はすべて公開しなければならない。
- プロジェクトの安定性は、開発者の間で徐々にできあがっていく集合知によってもたらされる。
- ボランティアの人たちの真の行動原理を理解しておけば、うまくやっていけるでしょう。
- 複数の人間が資源を共有して作業を進めていく以上、政治的な話は避けることができない。誰かがアクションを起こせば誰かが影響を受ける。
- 新入りのメンバーが困っていることに気づいたら、それはガイドラインをきちんと設定すべき事案なのに、それがまだできていないという証拠。新入りが劣っているというわけではない。
- コミット権を与えるか否かを決める主要な条件は、技術力ではなくその人が正しい状況判断ができるかどうかです。技術は後で勉強すればいいのです。
- ボランティアの集まりにおいては、いわゆるソーシャルスキル、つまり「集団の中でうまくやっていく能力」は技術力と同じくらい重要で、技術が優れているだけでは私たちは彼をコミッターにはしないでしょう。
- ひとつの管理作業を2人に任せると、その2人の間のコミュニケーションというオーバーヘッドが発生します。上手く作業が進む場合もあるし、お互い責任をなすりつけあって時間が浪費される危険性もあるでしょう。
- マネージャーの仕事で重要なのは、「誰かが自分の担当分野の作業をしていることに気づいたら、その人がうまくやっていけるように自分の方針を伝える。そして、みんなの作業が競合しないようにする」ということ。
- プロジェクトには、パッチマネージャー、翻訳マネージャー、ドキュメントマネージャー、バグマネージャー、リリースマネージャーがいます。これらの役割というのはあくまでも責任の範囲に関するものであり、独占させるためのものではない。
- 平等主義や協力体制を壊してしまう縄張り主義の登場を防ぐために、多くのプロジェクトではソースファイルから作者名や保守担当者名を取り除く方式を採用している。
- ボランティアを管理することも一種の「技術」であるといえる。
- セキュリティ脆弱性の処理のしかたは、攻撃者がコミットログをチェックしている可能性があるので、、修正が完了するまではバグのことを口外せず、バグがあったことをアナウンスすると同時に修正プログラムを公開するようにする。
- 1日にメールが数百件以上飛び交うほどの大きなプロジェクトになると、ツールとしてのメーリングリストのシステムは破たんする。そして古参の開発者は質問の質が低いメールをノイズと感じるようになりコミュニティを去ってしまい新参者だけが残る。
- たくさんの意見が行きかい「誰がいつ何を言ったのか」を追いかけるのが面倒になってしまうということ知っていて、あえて長文を送り付け議事進行妨害をする口やかましい少数派がいることがあるので気をつける。
- 初期段階で開発言語を決めるとき、それが他の言語より優れているからではなく、それがいちばん書き慣れているからという理由で選択すると、宗教戦争に陥らずに済む。
- ハッカー界、そして特にフリーソフトウェア文化においては、wonderful hackerのような肩書きに頼る人は排他的な臆病者とみなされます。
- 競合するプロジェクトについて否定的な意見を述べるのはやめる。理由は1.建設的でないフレームウォーが起こりがち2.将来競合プロジェクトで働く人が出る可能性がある3.競合相手が実際に優れている点を無視し客観的な判断ができなくなるから。

- 口コミによるマーケティングも無視はできない。
- コミュニティにお金を払ったとしても自分が望んだ機能をつけてもらえるとは限らないし、逆にボランティアのモチベーションが下がりプロジェクトから去ってしまうこともある。
- 投票は問題解決の1つの手段ではあるが、その瞬間から創造的な考えをやめさせてしまうことになるので、問題解決のあらゆる議論や手段が全てなくなるまで安易に投票に頼ってはいけない。
- オープンソースプロジェクトは複製すればすぐ別のプロジェクトにすることが容易なので全てを決定する独裁者は存在せず、民主主義的な意思決定のしかたをする。同様にプロジェクトは分裂しやすい。

index