*BSD プロジェクトにおけるドキュメント管理

佐藤 広生@東京理科大学

hrs@allbsd.org

The FreeBSD Project, Documentation Engineering Team
The FreeBSD Project, Release Engineering Team
The NetBSD Project
The DragonFly BSD Project

BSD **なひととき**, Networld+Interop 2005, June 9th, Chiba, Japan

おはなしすること

- **ドキュメントとは?**
- **現在の** *BSD **プロジェクト** (FreeBSD, NetBSD, OpenBSD, DragonFly BSD) **における ドキュメント管理体制**
  - **現状とポリシの比較**
  - **管理に用いられている技術と課題**
- **翻訳への取り組みと工夫**

資料はこちら:
http://people.allbsd.org/ hrs/N+I2005/sato-n+i.pdf
http://people.allbsd.org/ hrs/N+I2005/sato-bsdcon.pdf

ドキュメントとは

ドキュメントとは?

- ソフトウェアのインストール方法や使い方を書いた説明書。商用製品には必ず付いている。
- いわゆるフリーソフトウェアでは、ASCII テキストによる説明書が同梱されることが多い。
  → README ファイルや INSTALL ファイル
- もちろん、README や INSTALL だけではない。よく使われるソフトウェアや、複雑なソフトウェアの場合、さらに多くのドキュメントが作成されることがある。

ドキュメントとは

ドキュメントの種類と特徴 (1)

- README, INSTALL
  - ASCII テキストであることが多い。文章構成はさまざま。
- マニュアルページ: UNIX 系 OS で伝統的に用いられているドキュメント。
  - 形式は roff (テキスト整形ソフトウェアの一種)。専用のブラウザ (man(1) コマンド) を使う。

ドキュメントとは

ドキュメントの種類と特徴 (2)

- FAQ (Frequently Asked Questions): Q&A **形式の問答集。**
  - **問答の単位で独立しているので、複数の人が協力して作成するのが比較的簡単。**
  - Usenet FAQ **などが有名。**
  - **古くなった情報の保守が難しい。**

ドキュメントとは

ドキュメントの種類と特徴 (3)

- How-To: **特定の話題に特化したチュートリアル**
  - **柔軟な設定が可能な複雑なソフトウェアや、ソフトウェアを組み合わせて使う方法などを取り扱う。**
    ipfilter, sendmail, Samba, CVS **など。**
  - Linux How-To **などが有名。**
  - **ユーザのニーズが高いが、ソフトウェアの更新や世情の変化によって陳腐化しやすい。**
  - **雑誌記事として通用するレベル。**

ドキュメントとは

ドキュメントの種類と特徴 (4)

- **ハンドブック**: How-To **を系統的にまとめた網羅的なチュートリアル**
  - **規模が大きく、保守には労力を必要とする。**
  - **目次や索引作成といった編集作業の比重が大きい。**
  - **書籍として通用するレベル。**

ドキュメントとは

ドキュメントの種類と特徴 (5)

	文章量	書き手の負荷	編集負荷	共同作業	読みやすさ
ASCII テキスト	小	小	低	×	低
マニュアルページ		roff の知識	低	×	中
FAQ		小	中	○	中
How-To		文章構成力	中	△	高
ハンドブック	大	文章構成力	高	△	高

- **書き手の負荷・編集負荷・読みやすさのバランスは、** FAQ **が最も優れている。**
- FAQ **を拡充することで** How-To, **ハンドブックの形式へ移行する例も多い。**

ドキュメントとは

- **提供フォーマット**
  - **文書をどのような形で 提供 するか。**
  - **読み手の可読性に影響する。**
  - ASCII **テキスト、**HTML、PDF **が現在の主流**
- **管理フォーマット**
  - **文書をどのような形で 管理 するか。**
  - **書き手・編集者の保守効率に影響する。**
  - ASCII **テキスト、**roff、GNU info、HTML、SGML/XML、**独自形式など、さまざま。**

ドキュメントとは

- 提供フォーマットはあまり悩むところがない。
- 管理フォーマットは、目的とするドキュメントの種類によって適切に選択しないと、管理の負荷が高くなってしまう。
  - 書き手や編集者はそれぞれ、管理フォーマットの知識を持っていなければならない。
  - フォーマットによっては、表現が制限されることがある。
  - 目次や索引の作成など、機械的に実現できる部分に労力を投入すると破綻しやすい。

ドキュメントとは

ドキュメントの品質評価

- コードの品質は気にする人でも、ドキュメントの品質を気にする人は少ない。
  - コードは動くかどうかという絶対基準がある。
  - コードは美しいとか美しくないという議論があるくせに、ドキュメントは古くなっても「ないよりまし」という意見がまかりとおることがある。
  - 文章の正誤、構成の良否 → 文章としての品質
  - 規格適合性、保守コスト → 技術的な側面からの評価

ドキュメントとは

まとめ

- ドキュメントの種類
- 提供フォーマットと管理フォーマット
- ドキュメントの品質評価

FreeBSD の場合(1)

- **管理フォーマットに全面的に** DocBook/SGML **を採用。ほとんどのドキュメントが** ASCII, HTML, PDF **のフォーマットで提供されている。**
- **管理しているドキュメント**
  - **リリース文書** (**リリースノート、インストールガイド、ハードウェアノートなど**)
  - **チュートリアル** (FreeBSD Articles, **現在** 43 **本**)
  - **ハンドブック** (FreeBSD Handbook **他**, **現在** 11 **本**)
  - WWW **サイト**: www.FreeBSD.org (doc/ **と** www/)

FreeBSD の場合(2)

管理フォーマット: DocBook/SGML

- DocBook **と呼ばれる** SGML DTD **を用いて、ドキュメントをマークアップしている。採用は** 1998 **年と早く、**LinuxDoc DTD **から移行。**
- SGML **ドキュメントは、**DSSSL **スタイルシートを使って提供フォーマットに変換される。処理系は** Jade**。**

FreeBSD の場合(2)

管理フォーマット: DocBook/SGML (cont'd)

- ASCII テキスト、HTML、PDF 形式が同じ SGML ドキュメントから生成可能。
- www.FreeBSD.org にあるウェブページの一部には採用していない。
  - HTML/SGML を採用。移行や書き手の知識の問題。
  - 1 ファイル = 1 文書という対応があり、提供フォーマットに多様性が要求されない場合、SGML にするメリットは小さい。

FreeBSD の場合(3)

DocBook/SGML を使う利点と欠点

- 提供フォーマットが充実していて、それぞれを別個に準備する必要がない。
- 文書構造に一貫性を持たせることができる。
- 目次や索引の作成や、複数のドキュメントで共有する情報をひとつの場所に集めて一括管理するといった自動処理も可能。

FreeBSD の場合(4)

DocBook/SGML を使う利点と欠点 (cont'd)

- **書き手に** SGML **の知識を要求する。** SGML **の知識がないと文章を書いたり、修正・編集することが難しい。**
- **提供フォーマットと管理フォーマットの形が大きく違うため、たとえば誤字ひとつとっても、知識がないとどの部分を修正すれば良いかという判断が困難なケースがある。**
- SGML **に関する資料は非常に少なく、提供フォーマットに変換するツールが英語以外に対応していないものがある。**

FreeBSD の場合(5)

管理主体

- The FreeBSD Project **のドキュメント管理は、**
  - **リリース文書を**
    Release Engineering Team (re) **が、**
  - **その他の文書を**
    Documentation Engineering Team (doceng) **が**

それぞれ行なっている。

FreeBSD の場合(6)

現状と課題

- DocBook/SGML の採用により、10 年近く一貫性の高いドキュメント構造を維持できている。
- FreeBSD Handbook は、書籍としての出版もされている。目次、索引、TM マークの追加など、商業出版に要求される作業はすべて DSSSL スタイルシートのレベルで実現されている。
- SGML の知識が要求されるため書き手や編集の負担が大きく、ドキュメントにさまざまなルールが存在するため、貢献者が育ちにくい。

NetBSD の場合(1)

- 管理フォーマットには、.list という独自形式を用いていた。専用の Perl スクリプトを使って、ASCII, HTML のフォーマットに変換可能。
- 2003 年から全面的に DocBook/XML を採用。現在、www.NetBSD.org を含むすべてのドキュメントを DocBook/XML で書き直す作業が進行中。

NetBSD の場合(2)

- **提供フォーマットは基本的に** ASCII, HTML **であり、ハンドブック形式のもののみ、**ASCII, HTML, PDF **のフォーマットで提供されている。**
- **管理しているドキュメント**
  - **リリース文書**
  - FAQ **とチュートリアル** (FAQs and How-Tos, **現在** 71 **本**)
  - **ハンドブック** (NetBSD Guide, pkgsrc Guide, **現在** 2 **本**)
  - WWW **サイト**: www.NetBSD.org (htdocs/)

NetBSD の場合(3)

管理フォーマット: DocBook/XML

- DocBook/SGML **の** XML **版。**
- **提供フォーマットへの変換には、**XSLT **スタイルシートを用いる。処理系は** GNOME **プロジェクトで使われている** libxslt **をベースにした** xsltproc **というもの。**
- PDF **への変換は** FreeBSD **と同じ** DSSSL **スタイルシートを用いている。**

NetBSD の場合(4)

管理主体

- The NetBSD Foundation **には、**Board, EC (Executive Committee), PMC (Project Management Committee) **という組織体系がある。**
- **ドキュメントの管理は、**PMC **のひとつである** www-pmc **が担っている。ドキュメント管理における最終意志決定や、問い合わせなどを取り扱う。**

NetBSD の場合(4)

現状と課題

- XML は新しい規格であり、正しい知識を持っている開発者や、XML ドキュメントの処理に使えるツールが限られている。
- 管理フォーマットで書かれたファイルが、ソースツリーに一貫性なく格納されており、把握が困難。大量の文書がばらばらに置かれている状態。
- 今後、ドキュメントを整頓していくことが大きな課題。

OpenBSD の場合(1)

- 管理フォーマットは ASCII テキストと HTML であり、基本的に管理フォーマットと提供フォーマットが一致している。
- FAQ が多く、ハンドブックに相当する大規模な文書はない。
- 管理しているドキュメント
  - リリース文書
  - FAQ とチュートリアル (FAQ, 現在 11 本)
  - WWW サイト: www.OpenBSD.org (www/)

OpenBSD の場合(2)

現状と課題

- 基本的に手書きの HTML を使っている。目次の作成なども、すべて手作業。
- PDF 版も存在するが、HTML ブラウザを使って印刷イメージを PDF 化している。
- 管理手法に工夫している点は少ない。組織上に明確な管理主体は存在せず、ドキュメントを担当している開発者が数人いるのみ。

DragonFly の場合

- 管理フォーマットは HTML をベースにした独自形式のものと DocBook/XML が混在。
- HTML ベースの独自形式は、CGI で HTML に変換するようになっている。DocBook/XML は NetBSD から持ってきたもの。
- ドキュメントの書き手の負担を減らすために Wiki を使おうとしている。
- WWW サイト: www.dragonflybsd.org (docs/ と site/)

比較から分かること

	管理 fmt	**変換**	**提供** fmt
FreeBSD	SGML	DSSSL (Jade)	ASCII, HTML, PDF
NetBSD	XML	XSLT (xsltproc)	ASCII, HTML, PDF
OpenBSD	HTML, ASCII	**手作業**	HTML, PDF
DragonFly	XML, HTML	CGI, XSLT	HTML

- **ドキュメントの規模 (数・文章量) が増加してきた時に、管理フォーマットを効率の良いものに変更することは効果がある。**
- **ただし、書き手への要求の変化や変換ツールの依存などの問題に配慮する必要がある。**

日本語翻訳の取り組み

本当に必要ですか?

- 「日本語翻訳は信用ならないので英語を読めば良い」という意見と、「多少間違っていても、古くてもいいので日本語翻訳がほしい」という意見がある。
- そもそも日本・韓国・中国は英語の読み書きができる人が少ないが、特に日本は、アジアの中でも英語読解能力が低いことで有名。
- 必要とする人が大多数のはず。

日本語翻訳の取り組み

プロジェクト管理担当としての意見

- 品質を度外視する態度は歓迎されない。品質の低い翻訳は、プロジェクトとして受け入れることはできない。考え方はコードと同じ。品質を向上させる努力は必要。
- RFC の翻訳のように、「一度翻訳したらおわり」というものと、原文が更新され続けるものの違いを意識すること。

日本語翻訳の取り組み

プロジェクト管理担当としての意見 (cont'd)

- 原文が更新され続ける以上、翻訳は継続作業。継続して作業する強い意志が必要。
- 翻訳よりも編集作業のほうが大変。プロジェクトのドキュメント管理方法をきちんと学ばないと、翻訳を受け入れてくれません。
- 悲しいことですが、がんばっても誰もほめてくれません :P

日本語翻訳の取り組み

原文の管理主体との連係

- 一回翻訳したら終り、というものでない限り、必ず原文を管理しているところにコンタクトをとり、協力を要請すること。
- 自分がドキュメント管理に参加して、その上で翻訳をするという意識が必要。
  - 日本人は日本語のコミュニティをつくって何かしようとすることが多いが、それはあまり良い結果を生みません (し、向こうもあまり良い印象を持っていません)。

日本語翻訳の取り組み

ドキュメント管理における翻訳への配慮

- **ドキュメントを管理する側は、翻訳の存在を意識した枠組みを構築することが望ましい。**
  - **規格化・国際化された管理フォーマットを使う。たとえば** SGML **や** XML **は英語以外も対応可能。**
  - **ディレクトリ配置の工夫。「**doc/**言語コード/ソースファイル」のように、言語を意識する。**
  - LD(Language Dependent)/LI(Language Independent) **分離を行ない、共通化できる部分は共通化する。**

日本語翻訳の取り組み

翻訳への配慮: separate commit

- **原文の文章の内容変更と、形式的な変更を分離してもらう。これが混ざっていると、変更を追いかけるのが大変。**
- **分離していれば、英語に自信がない人でも形式的な変更を日本語に反映する作業ができる。**
- **翻訳する人は、より翻訳に集中できる。**

日本語翻訳の取り組み

翻訳への配慮: LD/LI 分離

- 開発者名簿やミラーサイトリストなど、原文と翻訳で共通化できるところを LI 部分として抽出し、重複作業を避ける。
- ドキュメント全体への総合的なアプローチ。

日本語翻訳の取り組み

LD/LI 分離: FreeBSD での実装例

- mirrors.xml: **ミラーサイトの情報をまとめたデータベース。**www.FreeBSD.org **や** FreeBSD Handbook **で共通に利用。**
- transtable.xml: **翻訳文書で** LI **部分を使う時に、日付や国名などを単語単位で自動翻訳するフレームワーク。**
- news.xsl: **新着情報が追加された時に、翻訳がなければ自動的に原文を翻訳に追加するフレームワーク。翻訳をあとから追加すると、自動的に翻訳が使われるようになる。**

日本語翻訳の取り組み

翻訳への配慮: LD/LI 分離 (cont'd)

- LD/LI 分離は、ドキュメント管理の負担を小さくするのにも有効。
- たとえば FreeBSD Hardware Notes は、マニュアルページの HARDWARE セクションから情報を抽出し、DocBook/SGML に変換して作成されている。
- デバイスドライバの開発者がマニュアルページを更新すれば、その情報がすぐに反映されるので、保守が容易。

まとめ(1)

- *BSD プロジェクトにおけるドキュメント管理は、規模の増大や要求される品質が高くなってきたことから、SGML や XML を採用している。
- ドキュメントには、管理の負荷、書き手の負荷、品質管理という側面からの評価が必要である。たとえば OpenBSD には分散作業が容易な FAQ 形式が多く、ほとんどを手書きで管理しているが、これも現実的な解のひとつである。

まとめ(2)

- 翻訳はドキュメントの一形態であり、原文の管理主体によって、翻訳に対する配慮・品質管理がなされるべきである。
- ドキュメントの LI/LD 分離は、保守の負担低減に大きな効果がある。ただし、ドキュメントが複雑化するため、注意深く設計しなければならない。
- コードのように方法論が議論されることはまれですが、ドキュメント管理にもいろいろ (地味な) 工夫があります。

将来的な課題

- docs.sun.com のような全文検索・ナビゲーションサイトの構築 (docs.sun.com は XML ベースの管理)
- リリースノートや Errata 文書、セキュリティ勧告など、即時性の高いドキュメントの取り扱い
- オーサリングや翻訳支援ツールの開発

おしまい

質問はありますか?