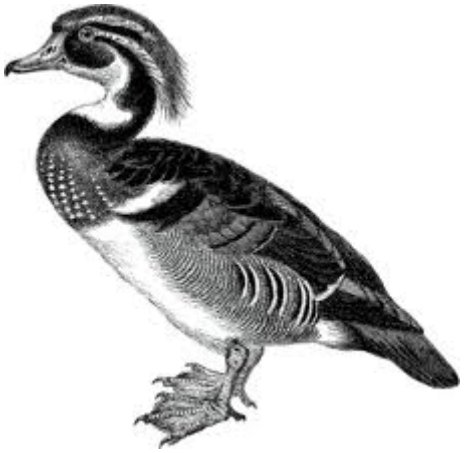
Общество с ограниченной
ответственностью «Название компании»



Разработка документации при помощи Docbook

[Пример к статье]

г.Нижний Новгород, 2014 г.

Содержание

Список иллюстраций

Глава 1. Предыстория

Так уж сложилось, что в наших проектах ведение технической документации полностью лежит на плечах разработчиков, по принципу: внес изменения в код проекта - актуализировал документацию. Сама документация представляла собой набор Word'овских документов, которая хранилась вместе с исходным кодом под VCS. Данный подход к организации разработки существовал долгое время, но пару лет назад мы решили озаботиться возможностью ведения документации проекта отличными от MS Office средствами.

Причин для этого было несколько:

- Первая, и, наверное, самая важная - частые конфликты при совместной правке файлов.
- Вторая причина – хотя документов было очень много, все они имели сходную структуру и во многом пересекались по содержанию (из-за особенностей архитектуры проекта). И как истинных программистов нас не устраивало "дублирование" текста.
- И наконец, вечная борьба со стилями оформления.

Все эти проблемы накладывались друг на друга и делали уже из этого, и так не очень любимого процесса актуализации документации, невыносимое по суровости наказание. Бывало что, после нескольких часов писанины, ты с чувством выполненного долга пытаешься "залить" свои изменения в SVN и получаешь нерадостное известие, что кто-то оказался быстрее тебя или ты просто напросто забыл обновиться перед началом работы. В любом случае это означало, что перекур придется немного отложить. По мимо текста необходимо было уделять внимание и стилям оформления, которые с довольно завидной регулярностью по каким-то причинам "ломались" (например нумераций списка началась с начала, в место того что бы продолжаться и т.п.). Причем не все такие "поломки" получалось легко устранить, иногда начинало казаться что Word живет какой-то своей жизнью и ему глубоко наплевать на твои пожелания к оформлению.

Таким образом, наша альтернатива MS Word должна была удовлетворять следующим критериям:

1. Текстовой формат хранения документа - для удобной работы с ним в VCS.
2. Поддержка широких возможностей оформления и стилизации документа.
3. Возможность декомпозиции конечного документа на фрагменты - для повторного использования.
4. Возможность публикации конечного документа в различных форматах.

В результате продолжительного поиска мы поняли что, решений, удовлетворяющим нашим требованиям, существует не так уж и много: DITA и DocBook.

DITA сразу нам показался слишком «мощным» и сложным для перехода, а вот на DocBook мы решили остановится. Вообще говоря, поиск альтернативного решения был очень постепенным и прежде чем мы поняли что "так жить дальше нельзя" и полным переходом на DocBook - прошел не один день и было проведено большое количество экспериментов над тем что было в тот момент у нас на руках. Первым делом попробовали хранить документы в формате WordML, что в какой-то степени решило проблему со слиянием изменений - теперь слияние не всегда заканчивалось конфликтом, но ручное разрешение конфликтов в разметке было очень дискомфортнным. Также пробовали разделить документы на фрагменты, тем самым снизив возможность конфликтных изменений и попытаться осуществить их повторное использование. Затея оказалась очень не удачной. И так постепенно, путем проб и ошибок, все таки решили полностью перейти на DocBook, так как по нашему мнению, он должен был устранить все наши проблемы.

Глава 2. Что такое «DocBook»?

Ели вдруг, кто не в курсе, *DocBook* - это стандарт описания документа и ничего полезного, кроме как, стандартизации содержимого он не делает. Причем стандарт достаточно старый, и многими, почему-то, считается уже устаревшим.

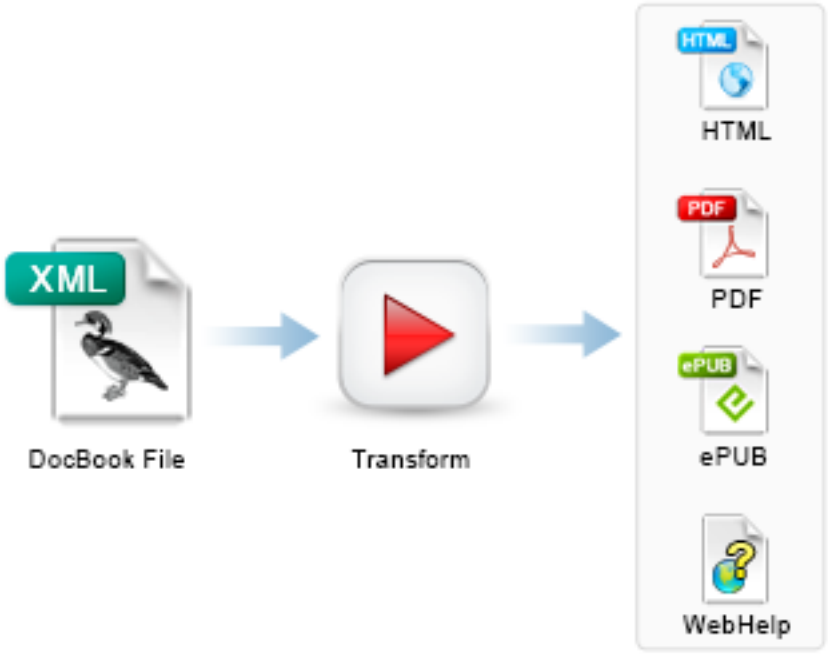
Написание документа в формате DocBook очень напоминает работу с HTML, только используется свой набор тегов и правил их употребления.

```
<book>
 <chapter>
 <title>First Chapter</title>
  <para>Hello world!</para>
 </chapter>
</book>
```

Данный пример демонстрирует описание книги состоящий из одной главы с названием "First Chapter" содержащий абзац с текстом "Hello Word!". Полный список тегов, а также примеры их применения можно подсмотреть на сайте проекта *www.docbook.org* . От себя хочу заметить, что набор тегов для описания содержимого очень (даже очень очень) большой, но в повседневной работе мы используем около 20.

Глава 3. Преобразование документа DocBook

Для того, что бы привести наш документ DocBook в какой-нибудь пригодный для чтения или печати формат необходимо использовать трансформатор (или даже конвейер из нескольких трансформаторов друг за другом), который на основе содержимого документа и, обычно, стилей оформления сформирует конечный документ.



Как правило для трансформации используется DocBook XSL (хотя существуют и более экзотические способы). Из коробки он уже поддерживает несколько форматов документов - html, xsl-fo, manpages и др. В случае, если требуется иной формат представления, то можно продолжить цепочку преобразований. Так для получения документа в PDF обычно используют следующую схему:

Рис. 3.1. Схема трансформации DocBook в pdf

И вот тут начинается самое интересное. Стили реализованные в DocBook-xsl по умолчанию позволяют получить нормальный по внешнему виду документ, но обычно, все равно требуется их кастомизация.

Разработчики стилей docbook-xsl позаботились об этой возможности и реализовали для этого специальные механизмы:

1. Самые общие параметры по формированию документа каждого из поддерживаемых форматов вынесены в отдельный файл param.xsl и для каждого из них есть более-менее подробное описание.

2. Существуют специальные шаблоны для формирования пользовательских шаблонов.

3. Наличие специальных, пустых по умолчанию шаблонов для последующего их переопределения.

Чаще всего, для управления процессом формирования документа разрабатывается собственный корневой XSL стиль, так называемый "Драйвер" в котором уже осуществляется тонкая настройка всех остальных параметров трансформации. Так как, каждый конечный формат в DocBook-xsl представлен своим набором шаблонов, то и "драйвер" под каждый из них нужно писать отдельный. Например, мы используем два конечных формата представления документа (xsl-fo и htmlhelp) и соответственно имеем два "драйвера" и два набора переопределенных стилей.

3.1. Выбор xslt и fo процессора

Для работы с DocBook-xsl необходим xslt процессор поддерживающий xslt версию 1.0. (Есть реализация docbook-xsl для версии 2.0, но не знаю на сколько она стабильная). На текущий момент существует множество рабочих решений под самые различные платформы - так что с этим проблем возникнуть не должно. В наших проектах мы используем saxon, правда старую версию - Saxon 9.1.0.8J, так как в последней бесплатной полностью убрана поддержка расширений EXSLT (необходимо для профайлинга документа) и не было уверенности, что расширение saxon для поддержки подсветки синтаксиса которое идет вместе со стилями будет работать в новой.

Для формирования документов из xsl-fo понадобится fo-процессор. Тут дела обстоят немного хуже - из рабочих процессоров лично я пробовал два FOP (opensource) и XEP (RenderX XEP Engine - немного платный). Существуют ещё несколько рабочих fo процессоров, но лично я с ними не пробовал работать и ничего сказать про них не могу.

Главный плюс FOP в том, что он бесплатный, но есть и минус - из "коробки" не поддерживает русский язык. При первом знакомстве с ним, нам так и не удалось заставить его работать с кириллицей. Как ни странно в интернете много статей про это, но все они были или очень старые (где предлагалось пересобрать FOP с нужными шрифтами) или содержали ошибки, которые не позволяли добиться нужного результата. В итоге все оказалось очень просто, но выбор наш уже пал на XEP. XEP прекрасно работает с кириллицей сразу же после установки и в принципе не требует какой-либо дополнительной настройки, но стоит 400$ - причем desktop версия. Различие в качестве рендера судить не берусь, но для интереса можете сравнить сами (в примере имеются файлы собранные обоими fo-процессорами).

3.2. Настройка стиля оформления

Для качественной настройки стилей необходимо немного знать язык xsl преобразования, а также язык разметки конечного документа. К сожалению, у нас в команде на момент перехода к DocBook такой компетенции не было и поэтому настройка заняла у нас достаточное количество времени - особенно для FO формата. Хотя в сети и существует большое количество сайтов с информацией по этому поводу (особенно ценный на мой взгляд "DocBook XSL: The Complete Guide") получить сразу полную картину очень сложно. Поэтому я решил поступить по принципу - "лучше один раз увидеть, чем сто раз услышать" и подготовил пример стиля для xsl-fo (примерно такого же, какой мы используем у себя в проектах) вместе с исходным текстом данной статьи и настроенным FOP.

Единственный момент, на котором хочу остановится и который по началу может сбить с толку - настройка шрифтов и языка документа. По умолчанию в xsl-fo включаются шрифты которые не поддерживают кириллицу, и если не переопределить эти параметры или допустить в них ошибку (необходимо убедится что fo процессор настроен на работу с указанным шрифтами), то на выходе из fo процессора скорее всего получим не читаемый документ. Язык документа влияет на создание автотекста для названий элементов книги (Глава, Книга и т.п). В принципе настройка только этих параметров уже позволит получить "корректный" документ. Также скорее всего возникнет желание поменять внешний вид титульной страницы документа. Сделать это можно при помощи специально подготовленного в docbook-xsl шаблона. Для этого необходимо определить свой вариант файла "/fo/titlepage.templates.xml" и при помощи xslt процессора и шаблона "/fo/titlepage.templates.xsl" получить настроенный стиль оформления для подключения в "драйвер". В некоторых случаях встроенных механизмов конфигурации не хватает, и тогда приходится уже явно переопределять в драйвере оригинальные стили docbook-xsl.

Глава 4. Заключение

Полный переход на DocBook у нас занял достаточно много времени. Во-первых нужно было привести к нему уже написанную документацию. Тут мы пробовали разные утилиты наподобие AntiWord, но из-за большого количества артефактов было принято решение сделать это вручную (артефакты получались как из-за ошибок форматирования в исходном документе, так и из-за особенностей скриптов перевода) . Также много времени отняла у нас разработка собственных стилей оформления, поиск среды для разработки документов (в итоге остановились на NotePad++) и настройки окружения. Казалось простая задача, но при её реализации постоянно натыкались на какие-то проблемы. К сожалению, не так много информации по DocBook, а если говорить о русскоязычном сегменте - то вообще практически нет. Но в итоге мы остались довольны.

С тех пор как наша команда перешла на ведение технической документации в DocBook прошел уже не один год, и мы уже не представляем для себя какой-либо инной вариант. Всего того, что мы хотели добиться переходом на DocBook - мы добились:

- Забыли что такое конфликты в документации при работе в VCS, даже при использовании GitFlow (с той же оговоркой, что и в начале статьи: сделал изменения - отразил в документации)

- Почти полностью избавились от дублирования одного и того же текста в разных документах. Благодаря профайлингу DocBook получилось сделать документ более гибким, а написание документации менее занудной работой. Тут главное чувство меры, так как через-чур сложная декомпозиция исходного документа сильно усложняет навигацию по нему и последующие редактирование.

- Практически забыли как форматировать документы в Word, а точнее сказать просто забыли как форматировать. Теперь разработка документации - только написание текста документа.

- Большой простор для творчества в плане интеграции в общий процесс разработки ПО.

Естественно кроме плюсов есть и минусы:

- Самая актуальная на текущий момент проблема - это взаимодействие с другими подразделениями компании. По понятным причинам только наша команда перешла на DocBook, все остальные так и используют MS Word, и когда нам нужно произвести взаимообмен данными - то приходится это делать в ручном режиме. Благо это бывает очень редко, и обычно ограничивается парой абзацев текста.

- Сложность реализации некоторых нестандартных для DocBook подходов к форматированию документов, в частности часто встает необходимость раз-

вернуть несколько страниц в landscape ориентацию, но вот до сих пор так и не научились этого делать (к нашему стыду) и приходится этот момент обходить как-то по другому. Но я на 100% уверен что такое сделать можно, и по этому нерешенность данной проблемы можно объяснить не столь острой необходимостью. К примеру, когда нам нужно было вставить формулы в документ - на прикрутку MathML ушло менее полудня.

А цель данной статьи в том, чтобы довести до читателей, которые ведут разработку технической документации в офисных программах, что существуют более подходящие инструменты. Для тех же, кто уже какое-то время посматривал в сторону DocBook или DITA, дать некоторый толчок и подсказки для перехода - ведь самое сложное начать! Так же, было бы очень интересно услышать какие подходы приняты в других командах и их опыт внедрения.

Список литературы:

- *http://docbook.sourceforge.net/*
- *DocBook XSL: The Complete Guide*
- *http://www.docbook.ru/*

Пример:

- *https://github.com*