

Mediawiki: silver bullet or swiss-army knife?

Stas Fomin

CustIS

www.custis.ru

email: stas@custis.ru

Abstract

General-purpose Wiki systems are successfully used in last years for documenting and knowledge management in software companies. These systems were designed to aggregate knowledge in open communities. But now Wiki systems often replace "stereotype" document management systems and software documenting technologies.

We will discuss some reasons for this situation and point out some problems that limit application scope of Wikis. Wiki is not a "silver bullet" but we can outline application domains where Wiki may be effectively used. Then, we will consider how to choose Wiki system in order to build an internal corporate knowledge base, keep and maintain documentation and communicate with customers.

MediaWiki is an optimal choice for this purpose in our opinion, since it has a vast community of users and developers and a wealth of experience. Open architecture makes it possible to extend easily any functionality of MediaWiki and to "downsize" possible inconveniencies.

Finally, we will offer solutions for the most acute problems of "corporate wikiing".

Keywords: *authoring, documenting, knowledge management, Wiki systems, MediaWiki.*

Mediawiki: Серебряная пуля или швейцарский нож?

Стас Фомин

Заказные ИнформСистемы

www.custis.ru

email: stas@custis.ru

Аннотация

В последние годы наблюдается бурный рост использования Wiki-систем, причем не только для агрегации знаний открытыми сообществами, для чего они были предназначены изначально, но и для процессов документирования и управления знаниями внутри компаний-разработчиков ПО. При этом, вики-системы зачастую заменяют «классические» системы документооборота и технологии генерации документации. Здесь мы выделим и проанализируем причины, по которым это происходит, и обозначим границы применимости — покажем минусы вики-систем, из-за которых они так и не стали «серебряной пулей», и выявим области, где использование вики-систем не только модно, но и действительно эффективно.

Далее, мы перейдем к вопросу выбора вики-системы для задачи построения внутрифирменной базы знаний, ведения документации и коммуникации с потребителями. На наш взгляд, в большинстве случаев оптимальным выбором будет MediaWiki. У этой вики-системы огромное количество пользователей и разработчиков, накоплен опыт использования и объем текстов. Открытая архитектура системы позволяет легко расширять возможности инструментами-расширениями и «срезать» неудобные углы в использовании.

И наконец, мы рассмотрим наиболее острые проблемы, встающие перед пользователями вики-систем и предложим свои решения.

Ключевые слова: *документирование, коммуникация, управление знаниями, вики-системы, MediaWiki.*

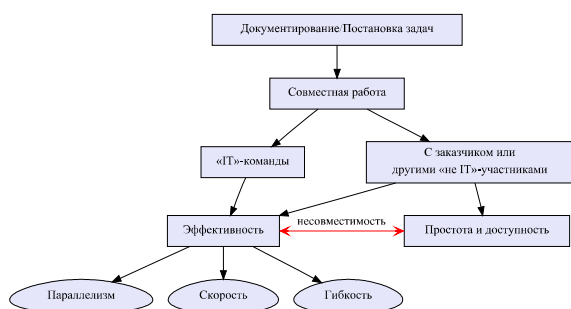
1. Задачи

Рассмотрим по очереди задачи, возникающие в любой компании разработчике ПО:

1. эффективное документирование и взаимодействие с заказчиком или потребителем;
2. построение внутренней информационной базы знаний.

1.1. Документирование и взаимодействие

Ранее было принято, что технической документацией занимается небольшая группа технических экспертов (технических писателей), владеющих технологиями верстки, публикации и групповой работы. Сейчас же, с ростом использования Agile-практик разработки, стало критически важно привлекать к работам по постановке задач или рецензированию существующей документации максимально широкий круг специалистов, как внутри компании (аналитиков, тестировщиков, разработчиков), так и представителей заказчика. Качество верстки, полиграфического оформления и литературного стиля уже не так важно, как актуальность, своевременность, полнота учета интересов всех заинтересованных сторон и верифицированность различными специалистами.



Т.е. важно, чтобы используемая технология обеспечивала возможность совместной работы, но для различных групп участников важными являются разные и конфликтующие между собой аспекты. Так, например, для основных или стержневых производителей документации, обозначенных на рисунке как «IT-команда», ключевым является фактор *эффективности*, выражающийся в следующих аспектах:

Параллелизм — возможность асинхронной работы с минимумом блокировок и других узких мест, что дает возможность масштабировать команду. Правильными практиками для этого аспекта являются использование систем контроля версий с *неблокирующей* моделью «Копирование-Изменение-Слияние», допускающей параллельную правку одного артефакта несколькими участниками. Отрицательными примерами могут служить схемы «согласование по email», «файлы на

файл-сервере с блокировкой», «Word-документ на всех».

Скорость — аспект технологии, ответственный за минимизацию затрат (человеко-часов, килокалорий, меганейронов) на условную единицу документации. Позитивные примеры — «написал несколько коротких строчек — диаграммы сами нарисовались, текст сам красиво отформатировался». Отрицательные — «выравнивать мышью маловажные диаграммы», «мучительно и монотонно переформатировать текст».

Гибкость — цена внесения изменений должна быть ограничена и не более чем пропорциональна объему изменений. Хорошо, когда используются различные составные документы, шаблоны, автоматическое построение документации по программному коду (*literate programming*) или по схеме базы данных. Плохо, когда изменения надо проносить вручную и во множество различных документов.

С другой стороны, кроме «стержневых» участников процесса, обеспечивающих основной объем документации, не менее важна роль и остальных — представителей заказчика, бизнес-аналитиков, разово привлекаемых экспертов-рецензентов. *Можно привести смелую метафору с «вытягиванием репки» — несмотря на то, что основную тяговую силу составляет стержневая команда «деда и бабки», без участия «кошки и жучки» уже не обойтись, а минутное участие «мышки» может спасти проект от тупика.* А для них в первую очередь важна не эффективность построения документации, а простота ее правки-рецензирования и доступность для изменений. *Простота* означает крутость «кривой обучения» — т.е. включиться в работу с допустимым качеством можно практически «не приходя в сознание», посмотрев пару примеров, и ориентируясь на интуицию. Как пример, можно сравнить программирование на Basic против программирования ядра операционных систем, или использование текстовых процессоров в сравнении с системами верстки типа TeX.

Доступность же означает, что к процессу можно «подключиться» откуда угодно, нет никаких ограничений и требований к рабочему месту, не требуется сложных инсталляций набора ПО. В идеале должна использоваться модель «тонкого» клиента, под которым в данный момент обычно подразумевают браузер.

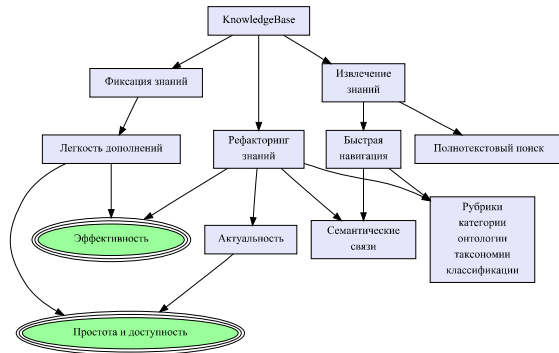
Как обычно, аспекты «эффективность» и «простота и доступность» плохо между собой совместимы, и решения выигрышные для одной стороны, обычно совершенно недопустимы для другой. Но компромисс необходим — иначе одна из сто-

рон («соге» или «не-соге») неизбежно выйдет из игры.

1.2. Knowledge management

Теперь рассмотрим задачу построения корпоративной базы знаний. Условно можно выделить следующие направления (см. рис.):

1. Фиксация новых знаний;
2. Рефакторинг знаний — поддержание актуальности, классификация, установление семантических связей и т.п.;
3. Извлечение знаний.



Оказывается, первые два пункта нуждаются в том же, что рассмотрено в предыдущем разделе, ибо с одной стороны требуется «простота и доступность» технологии для широкой аудитории — от программистов и системных администраторов, до HR и административно-хозяйственных служб, т.е. включая персонал, не сильно «разбирающийся в IT». С другой — эффективность, как добавления знаний, так и рефактинга, причем с учетом совместной работы.

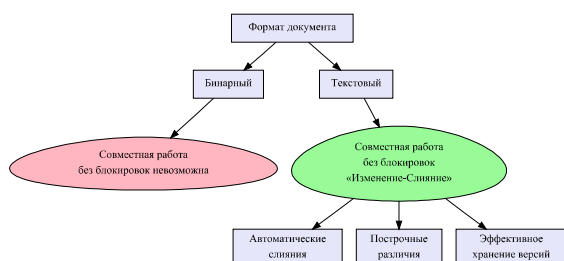
И наоборот, третий пункт — *эффективное извлечение знаний*, теперь востребован потребителем и от обычной документации. Ушли в прошлое времена, когда пользователь ПО должен был обязательно проштудировать увесистую стопку томов бумажной документации. Сейчас упор идет на возможность фрагментарного ознакомления, для чего тоже нужно предоставить все современные технологии — быструю навигацию по различным классификаторам, использование гипертекстовых переходов по семантическим связям, и эффективный полнотекстовый поиск с учетом морфологии.

Таким образом, видно, что задачи *Knowledge management*'а и документирования во взаимодействии с заказчиком весьма схожи, и для них разумно ожидать единого решения. Отложив вопрос, а существует ли такая «серебряная пуля», попробуем понять, каких свойств можно ждать от такого решения.

2. Ожидаемое решение

Формат документа

Сначала придется определиться с форматом информационных артефактов — статей и иных бло-

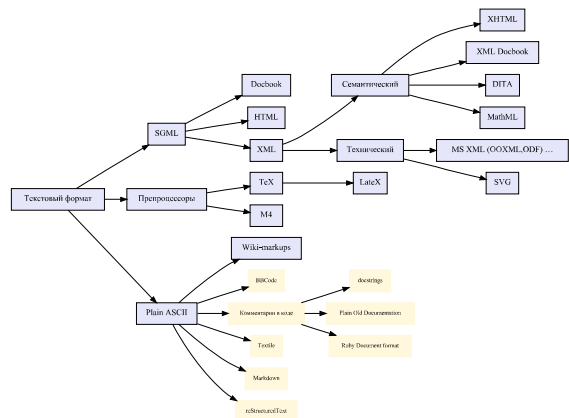


ков документации. Оказывается, с учетом необходимости эффективной совместной работы, нужно отказаться от (или по крайней мере минимизировать) использование артефактов в бинарных форматах.

Действительно, только для плоского, разбитого на ограниченные строки текста существуют эффективные алгоритмы вычисления различий и автоматического слияния различных версий. Этот момент уже известен с появления в 1970х первых систем управления версиями. И в эффективном документировании давно было принято использовать различные *языки разметки*, хранить документацию в текстовом виде под системой контроля версий, а бинарные форматы использовать только в неизбежных случаях, для редко меняющихся артефактов, например фотографий или растровых иллюстраций.

Языки разметки

Выбрав текстовый формат документа, какой же язык разметки выбрать? Мы возьмем на себя смелость предложить некоторую классификацию существующих языков (*markup languages*) разметки документов.



Не вдаваясь в историю и тонкости, отметим, что ветви языков разметки, основанных как на использовании препроцессора (TeX, LaTeX), так и на использовании жестких SGML-спецификаций (HTML, DocBook) обладают особенностями, делающими их неподходящими для решения задачи «простота для не-гиков». Элементы разметки, будь то теги или макросы, занимают много места, разметка документа ориентируется не на удобство автора, а на удобство машинного представления, для обработки препроцессором (TeX/LaTeX) или стандартизированным компилятором (SGML/XML/...). В результате, надо быть готовым к трудноуловимым и даже необъяснимым ошибкам при работе с TeX, или держать в голове иерархию вложенности сотен тегов при использовании структурной SGML/XML-разметки (DocBook, XHTML). Конечно, сильная команда легко сможет обуздать любое из этих средств, и сможет эффективно порождать документацию на любом языке разметки, используя системы последова-

тельных преобразований (*tool chains*), управляемые системой сборки (*make, Ant, Scons*). При этом можно достичь и *гибкости*, за счет использования препроцессирования, шаблонов и составных документов, и *параллелизма*, используя систему контроля версий (*SVN*, или хотя бы *CVS*).

Но «простоты» при этом уже не будет, ибо приучить «непродвинутого» человека этим пользоваться совершенно нереально — проверено более чем десятилетним опытом автора статьи, регулярно ставившего подобные эксперименты. Плюс необходимость развертывания системы сборки на каждом рабочем месте существенно подрывает «доступность».

Однако в те же 1970-80ые года, когда зародились поколения *SGML* и *TeX*-разметок, появились и так называемые плоские (*plain text*) разметки, основной смысл которых заключался в форматировании документа с помощью пустых строк, пробелов-отступов, и использования десятка символов пунктуации так, что текст оставался вполне читаемым и без обработки.

Получается, что «при всем богатстве выбора другой альтернативы нет» — наша «серебряная пуля» должна использовать текстовый формат документа и понятную *plain text*-разметку. При этом мы уже не достигнем качества верстки *LaTeX*-текстов, или удобства манипуляции *XML*-документами, но для поставленных задач — база знаний или техническая документация этого будет достаточно.

Другие желания

Что же еще разумно пожелать от «серебряной пули»? *Доступность* — просмотр гипертекста в любимом браузере, правка и редактирование там же. Да, ходить по ссылкам сможет и ребенок, но надо сделать, чтобы и ссылаться можно было легко и понятно. Чтобы было лекарство от страха — управление версиями. Чтобы обеспечивалось *reusability*: составные документы, шаблоны, препроцессор. В идеале, возможность подключения и использования любых простых предметно-ориентированных языков (*Domain Specific Languages*) для увеличения продуктивности. Например, построение графов и даже *UML*-диаграмм по краткому текстовому описанию, понимание математических формул в общепринятой разметке *TeX/LaTeX*, автоматическое форматирование и синтаксическая раскраска для фрагментов кода и т.п. Ну и конечно средства поиска и навигации — категории (онтологии, таксономии) и полнотекстовый поиск.

3. Вики-системы

Хотя «шум» вокруг вики-систем усилился в последние лет пять, на самом деле идея и первоначальная реализация появилась еще в 1995 году. Тогда разработчик общедоступной базы знаний по паттернам программирования, решил заложить простоту, общедоступность и эффектив-

ность в основу своей CMS-системы. Более того, он назвал ее в честь понравившихся ему гавайских автобусов, которые, несмотря на некоторую неказистость, дешево и эффективно перевозили пассажиров между аэропортами. Некоторое время вики-системы не были особенно заметны на фоне огромного множества других CMS-систем, растущих, как грибы после дождя. Но уже в 2001 году на вики-системе стартовала «Википедия» — теперь уже всемирно известная многоязычная энциклопедия, а в 2007 году слово «Wiki» получило официальную прописку в *Oxford English Dictionary*. Теперь практически базы знаний сообществ сделаны на вики-системах, на «виках» ведут официальные сайты проектов, «вики» попадают и в интранет, где вытесняют «классические» CMS и системы документооборота, в них же ведут техническую документацию к программным продуктам. Почему это происходит? Видимо, потому, что множество вышеперечисленных «хотелок» прекрасно покрывают базовые принципы вики-систем:

1. Простой язык разметки;
2. Совместное редактирование;
3. Правка и публикация по месту;
4. Централизованное хранение и версияльность;
5. Упрощенное построение ссылок.

Поясним пункты с третьего по пятый. Публикация по месту позволяет вносить правки в процессе чтения материала. Т.е. из режима «чтение» можно немедленно перейти в режим «писатель/корректор», и не нужно искать исходные тексты. Немедленная публикация позволяет сразу же проверить внесенные правки.

Также поясним, что при работе с файлами в «стандартных» языках разметки (*TeX, LaTeX, SGML Docbook, HTML*), нужно помнить нетривиальные соответствия между идентификаторами и названиями структурных блоков — секций, глав, разделов. Плюс еще помнить, в каком файле что лежит. Это конечно способствует целостности, но вносит огромную нагрузку на внесение ссылки. Не говоря уж, что для грамотной работы с файлами обязательно нужна система контроля версий — и все это минусы к «простоте и доступности». В виках же реализовано централизованное хранение всех блоков текста (документов, статей), и для них эквивалентны понятия идентификаторов, названий, и заголовков. Дополнительная дружелюбная к пользователю «денормализация» состоит в перенаправлениях ссылок и даже «опережающих» ссылках на несуществующие статьи.

Отлично подходит для компаний и социальные свойства «вик»:

- Никто не знает всего, но возможно собрать знания «с миру по нитке»;

- Никто не застрахован от ошибки, но любой, заметив ошибку, может легко ее исправить;
- Легче поддерживать актуальность — правка ошибки очень проста, и можно не бояться сломать — есть контроль версий;
- Совместное редактирование влечет совместную ответственность, вырабатывает культуру обсуждений и поиска правильного решения;
- Эффект «взбивания сливок» — легкость совместного редактирования ведет к многократным итерациям и улучшает качество;
- Легкость порождения статей способствует фиксации больших объемов знаний («главное — начать»).

Более того, очень многие проблемы открытых вики-систем, внутри компании перестают быть таковыми. «Спам и вандализм?» — поможет служба кадров. «Голпа невежд лоббируют ерунду?» — Поздравляем, обнаружена серьезная проблема компании, обязательно требующая социального решения. «Неразрешимые флеймы?» — Немедленно перейти к устным переговорам до компромисса. «Расстраивает потеря авторства?» — Похоже, кто-то не любит работать в команде...

Да, есть и технические проблемы, ограничивающие область применимости «вик», о чем надо помнить. Например, если вам нужна книга с высококачественной версткой или полиграфией (шрифты, сложные страницы с полями, плавающие объекты, оптимальный кернинг и выравнивание пустых пространств) — то я бы рекомендовал использовать LaTeX. Если заказчик требует сдавать документацию по древнему ГОСТу, с обязательной проверкой форматирования «службой нормоконтроля» — в таких случаях мы используем SGML DocBook. Но вообще тенденция такова, что ценность бумажной технической документации уже упала почти до нуля, и продолжает падать дальше. Правда есть еще область электронных документов, в которых требуется качественная верстка по «листу/экрану» — это презентации. Здесь я бы рекомендовал LaTeX/beamer, и использовать все возможности автогенерации текста, диаграмм, картинок, и автоматизировать все это системой сборки типа SCons. Но в целом, пока кажется что эта пуля, если и не тянет на «серебряную», то «посеребряной» называться вполне может. И самое главное, кроме словесных аргументов есть убедительная проверка этого подхода практикой. Почти все знают «Википедию», но далеко не все знают, что ее прародителем был проект «Nupedia», основанный на «серьезных и глубоко продуманных подходах». Например, процесс помещения статей в энциклопедию состоял из семи этапов (проверка независимыми экспертами, вычитка редакторами и т.п.). Результат за 4 года работы — 23 завершённых статьи, 68 «in

progress». А с момента перехода проекта на первый вики-движок, начался настоящий «взлет ракетой»:

2004: ≈ 300 000 статей («en»-раздел);

2005: ≈ 500 000 статей («en»-раздел);

2007: ≈ 2 млн. статей («en»-раздел).

В момент написания этой статьи (30 июля 2008 г.) в англоязычном разделе было 2,478,333 статей, а в русскоязычном 303,462. Для сравнения — последняя (v15) версия энциклопедии «Британника» содержала 120 тыс. статей, а последняя (v3) версия БСЭ состояла из 95 тыс. статей.

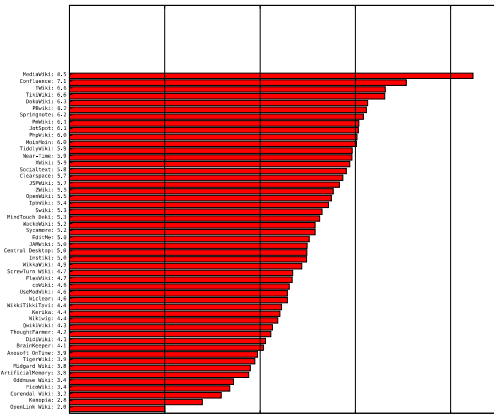
4. Обратная сторона луны или проблемы выбора

Итак, т.к. большинство читателей к этому моменту воскликнут «Убедили! Дайте две!», пора рассказать об обратной стороне, серьезно затрудняющей выбор. Дело в том, что вики-систем, даже только тех, которых сравнивают на *wikimatrix.org*, уже больше сотни, и их число продолжает расти. Почти у всех из них свой собственный язык разметки и интерфейс. Да, наступило время веб-технологий и сменился парк изобретаемых программистами «велосипедов» — с текстовых редакторов и языков программирования, на вики-системы и языки разметки. Выбрать же очень не просто — дискуссии по выбору оптимальной вики-системы, судя по нашему опыту, чудовищно жаркие — сравнивается функционал, эстетика, архитектура, в ход идут религиозные доводы («hate Perl!», «hate PHP», «болд обозначать астерисками!»...). И ошибка в выборе может надолго, если не навсегда, отпугнуть компанию от использования вики-систем: «Вики? Даже не предлагайте! Мы тут пробовали — кошмар, убогая, неудобная, падает, помойка, ничего найти нельзя, никто ее не знает, вообще потом сломалась и мы все потеряли». Ситуация, как в анекдоте — «Ка-рузо? Полная ерунда, мне вчера сосед напел...».

Чтобы этого не случилось, мы предлагаем в первую очередь ориентироваться на комплексную характеристику функционала системы — популярность и распространенность. Да, «миллионы леммингов не могут ошибаться»¹.

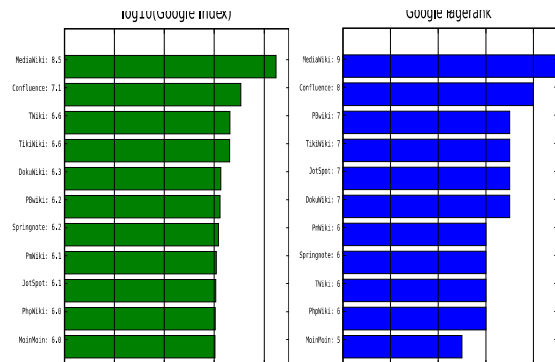
Как же объективно измерить популярность?

¹ Если кто не в курсе, популярный миф про леммингов, тупо следующих друг за другом на пути к суициду — всего лишь мем, запущенный постановочной сценой фильма «Белая пу-стошь» (1958), где леммингов сбрасывали в реку метлой.



Мы провели некоторое оригинальное исследование, подсчитав количество страниц в «глобальном интернете», т.е. индексе Google, упоминающих ту или иную систему. Диапазон полученных значений был настолько велик (от 101 до 295,000,000 страниц), что на графике и в последующем сравнении мы приводим десятичный логарифм от этого количества.

Видно, что популярность уже далеко не одинакова, и чтобы выбрать более-менее известную систему, желательно сразу ограничить «шорт-лист», верхней десяткой кандидатов, чтобы не оказаться в незавидной роли «лабораторных животных» — первых пользователей системы, самостоятельно занимающихся «разминированием» сложных моментов. Еще мы анализировали популярность, используя *Google Pagerank*, — известный индекс интернет-цитируемости. К нашему некоторому удивлению, «top10» согласно нашему исходному критерию и критерию, основанному на *Google Pagerank*, совпали! Из-за экономии места мы не приводим некоторые другие метрики популярности, например *Google Trends* и т.п. — заинтересовавшийся читатель вполне сможет получить их самостоятельно.



В целом, основная наша рекомендация — ограничить выбор первой вики-системы одной из популярных перечисленных. У каждой из них есть уникальные конкурентные преимущества, положительные и отрицательные особенности. Например, Confluence очень хорошо интегрируется с популярным баг-трекером Jira, но является

платной. TWiki бесплатна и поддерживает управление правами на статьи, но написана на Perl. MoinMoin понравится любителем Python'a. А JotSpot уже понравился Google, и его превратили в Google Sites.

Мы же продолжим аргументировать за наш выбор — *MediaWiki*. Количество контента, размещенного в MediaWik'ax, количество пользователей и разработчиков на порядки больше, чем у любой другой системы. Знание разметки MediaWiki уже не менее полезно, чем знание HTML — хотя бы потому, что дает возможность копировать или публиковать материалы в Википедии. Открытая архитектура имеет сотни возможных точек расширения (*Hooks, Extensions*), куда можно подключать свои модули, легко реализуя собственные предметно-ориентированные языки, дополнительные интерфейсы редактирования и публикации, поиска и навигации. Причем API-интерфейсы достаточно стабильны — т.е. реализовав расширение, можно спокойно обновлять MediaWiki, бесплатно получая новый функционал, без боязни потерять свои наработки. Более того, уже есть больше *тысячи* готовых расширений, которые легко может установить даже энтузиаст (скопировать файлы в каталог */extensions*). Остается лишь проблема выбора, но, как знают те, кому за тридцать, она гораздо приятней проблемы дефицита и отсутствия. По сути, именно система расширений превращает MediaWiki в многофункциональный швейцарский нож, причем в который вы легко можете добавить свои собственные инструменты. Перечислим несколько наших основных «штопоров», «напильников» и «пассатижей» из *swiss-army knife* «MediaWiki»:

Автоматическое построение графов — строит направленные и неориентированные графы по краткому текстовому описанию. Успешно используют даже самые далекие от компьютеров сотрудники компании и заказчиков. Подходят для всего — для рисования графов информационных и материальных потоков, диаграмм состояний или иерархии счетов, схем проводок или каких-либо классификаций. Кстати, все графы в этой статье построены именно по этой технологии.

UML-диаграммы по текстовому описанию — аналогично, автоматическое построение некоторых типов UML-диаграмм, но сформулированных на специальном текстовом языке *UMLGraph*, напоминающем описание классов на языке Java. Кстати, это совсем не профанация — *UMLGraph* рекомендовал к использованию Мартин Фаулер (Martin Fowler), один из самых известных популяризаторов UML.

Графики по тексту — двух- и трехмерные графики и диаграммы по записанным формулам или наборам данных. Да, редко кому в Software Engineering нужно рисовать функции Бесселя, но вот быстро опубликовать или обновить «*Scrum Burndown Chart*» бывает очень полезно.

Синтаксическая раскраска — для фрагментов кода или языков разметки. Быстро придает нарядный вид скучным фрагментам технической документации.

LaTeX — вставка формул и даже целых фрагментов LaTeX-документов.

Mindmaps — публикация так называемых «карт концепций», сделанных в популярной программе *Freemind*. По сути, это неограниченные иерархии понятий с взаимосвязями: планы действий, иерархии рисков, конспекты описаний предметных областей и т.п.

Экспорт в СНМ — экспорт содержимого вики-системы для последующей компиляции в статический СНМ-файл. Также есть экспорт в MS Word и OpenOffice.

MediawikiQuizzer — система тестов с выдачей электронных «дипломов», превращающих вики-систему в систему дистанционного обучения, ибо MediaWiki сама по себе уже подходит для публикации обучающего материала.

Полнотекстовый поиск с морфологией — используется отличный поисковик Sphinx (*sphinxsearch.com*), поддерживающий русскую морфологию. Кстати, для MediaWiki на PostgreSQL, скоро можно будет воспользоваться полнотекстовым поиском с русской морфологией, встроенным в ядро этой СУБД.

WikEd — встроенный в браузер JavaScript-редактор MediaWiki-разметки. Поддерживает синтаксическую подсветку и форматирование, поиск-замену, импорт из HTML или форматов офисных документов и т.п. Дает возможность начать работу без изучения вики-разметки, причем практически не чувствовать ограничений редактирования в браузере. Действительно, современные браузеры поддерживают и проверку орфографии, и закладки, и практически заменяют собой ранее популярные файловые навигаторы с многофункциональным редактором.

Викификатор — автоматический эвристический обработчик, улучшающий форматирование и типографику текста. Например, заменяет обычные «кавычки», т.е. знаки дюйма, на типографские кавычки-лапки, знак «минус» — на длинное тире, правильно расставляет пробелы около знаков пунктуации и т.п.

Впрочем, перечислять можно еще очень долго, поэтому для тех, кто заинтересовался, на конференции SEC(R)-2008 мы планируем раздавать переносимую версию MediaWiki с нашим набором расширений и демонстрационным набором статей, описывающих ее возможности. Любой желающий сможет простым копированием запустить вики-систему на своем рабочем компьютере или даже ноутбуке, продемонстрировать и попробовать систему в действии вместе со своими коллегами, и если всем понравится — перенести на сервер. Возможно, полезным будет и

личная инсталляция MediaWiki на ноутбуке, как персональная CMS или база знаний. Да, мы раздаем переносную систему под Windows, но те, у кого-то стоит Linux, мы уверены, самостоятельно справятся с инсталляцией.

И даже если MediaWiki вам не подойдет — вы сможете это быстро понять. Но знайте, что система быстро развивается, и лучше не заниматься бесплодными поисками «такого же, только без крыльев», а взять стандартное решение и подождать, пока не критичный функционал кто-нибудь сделает.

5. FAQ или наши ответы на часто задаваемые вопросы о «виках»

Здесь будут ответы на некоторые актуальные вопросы, о которых нас спрашивали в личном общении или на интернет-форумах.

Сразу скажем, что для внедрения и поддержания «вик» очень полезно, чтобы в компании был один или несколько энтузиастов, получающих удовольствие от «борьбы за качество». Поищите их среди технических писателей или тестировщиков. Отлично, если у них будет терпимый уровень грамотности и элементарные понятия о типографике. Поощряйте их участие в «модерации-викификации»! Вики — это не помойка, требующая регулярной уборки, это скорее сад, где регулярные усилия садовника направляют и придают растениям правильную форму.

Статьи в вики — это как «письмо из Простоквашино»? Нагромождение разных идей, стилей, «кто в лес, кто по дрова»?

А вот мы наоборот, читали, что у большинства статей даже в Википедии один или два выделенных автора, а остальные участники вносят пренебрежимо малые правки. В корпоративных виках все будет еще более авторским. Тогда зачем вики, где тут «синергия»?

Нет, ситуация «письма из Простоквашино» редка, почти у каждой статьи есть основные авторы, реализующие основной объем, а роль остальных сводится к правкам, дополнениям, замечаниям. И эта ситуация совершенно нормальна — вспомним приведенную метафору «Сказки о репке».

Есть у нас вики, только уже через год мы ничего в ней найти не можем. Полная помойка в ней образовалась, т.к. никто ничего не модерировал.

Обязательно сделайте возможность полнотекстового поиска, желательно с русской морфологией. Это более-менее спасает даже такую помойку, как интернет. Технически, если внутренний поиск в вашей вике плох, самое простое решение — установить *OmniFind Yahoo! Edition* (корпоративный бесплатный поисковик с русской морфологией). Чуть более сложно, т.е. требует некоторого понимания и доработки, подключение к вашей вике Sphinx (*sphinxsearch.com*). Не надо забывать об автоматических функциях Mediawiki по борь-

бе с нецелостностью и неактуальностью статей. Дополнительно мы используем LinkChecker (linkchecker.sourceforge.net).

Мы завели вику, но никто из сотрудников туда не пишет. Что мы делаем не так?

Надо выяснить, могут ли писать что-то сотрудники вообще. Может они ведут активную переписку с заказчиком и между собой по email, или даже, не дай бог, по ICQ. Покажите преимущества накопления знаний, по сравнению с использованием компьютера как телеграфа. Найдите тех, кому вики будет наиболее полезна, и помогите им начать, а остальные подтянутся. Может, сотрудники стесняются (например, из-за низкой грамотности) писать в общую вику — мотивируйте их не бояться и фиксировать любые полезные знания. Попробуйте и кнут — например, включить в *Definition of Done* по задачам, требующим отчета, отчет именно в вики-системе.

При обсуждении статей в MediaWiki сотрудники пишут комментарии не на вкладку «Обсуждение», а непосредственно в тело статьи. Как нам отучить их от этого, или получить «очищенный» от замечаний вариант?

Да, действительно контекстное замечание удобней вставить непосредственно в текст статьи. Так оно больше привлечет внимание, ведь если на выходе должен быть «чистый» текст, значит нужно рано или поздно учесть это замечание и удалить его. Оформляйте замечания в виде специальных шаблонов, которые с одной стороны будут привлекать внимание цветом или пиктограммами, с другой — не будут включаться в композицию статью. Используйте теги «noinclude».

Мы используем вики, но версии ведутся только для текстовых документов. Как хранить все версии и для бинарных объектов, например, картинок, чтобы использовать их также в вики.

Установите Subversion и храните бинарные объекты в нем. В любом случае для работы с нетекстовыми объектами вам понадобится специальные программы, вики обычно для этого не приспособлена (хотя есть некоторые расширения по редактированию картинок в браузере). Настройте доступ к SVN-репозитарию через Apache, и вы прекрасно сможете использовать картинки, и другие бинарные артефакты, ссылаясь на них из вики-системы. Кстати, вы также сможете ссылаться на любой программный код, документы в бинарных форматах, автогенерируемую по коду документацию и т.п.

Хотелось бы иметь возможность продолжать работу без онлайн-доступа, в местах без интернета, например, в командировке в регионы.

Заведите персональную инсталляцию MediaWiki (это легко и под Linux, и под Windows), перенесите экспортом набор статей и работайте над ними. Еще сейчас проходит обкатку реализация

протоколов *WebDav/DeltaV* для доступа к MediaWiki, что даст возможность использовать стандартный Subversion-клиент (например, TortoiseSVN), для экспорта статей, как набора файлов, редактирования их в файловом виде, с возможностью массовых правок, и последующем *commit*'е их в MediaWiki, как в SVN-репозиторий.

Мы перепробовали все, но вики-системы не приживаются. Что нам делать?

Радуйтесь! Значит они вам и не нужны! А если вы чувствуете, что они вам таки нужны, т.е. есть нерешенные проблемы в управлении знаниями или гибком документировании при взаимодействии с заказчиком — то проблема явно в другом. В людях и в их мотивации, в задачах компании и ее приоритетах. Да и вряд ли другой волшебный инструмент сможет помочь в этом случае.