Документирование ПО

Алексей Островский

Физико-технический учебно-научный центр НАН Украины

26 марта 2015 г.

Документация на ПО

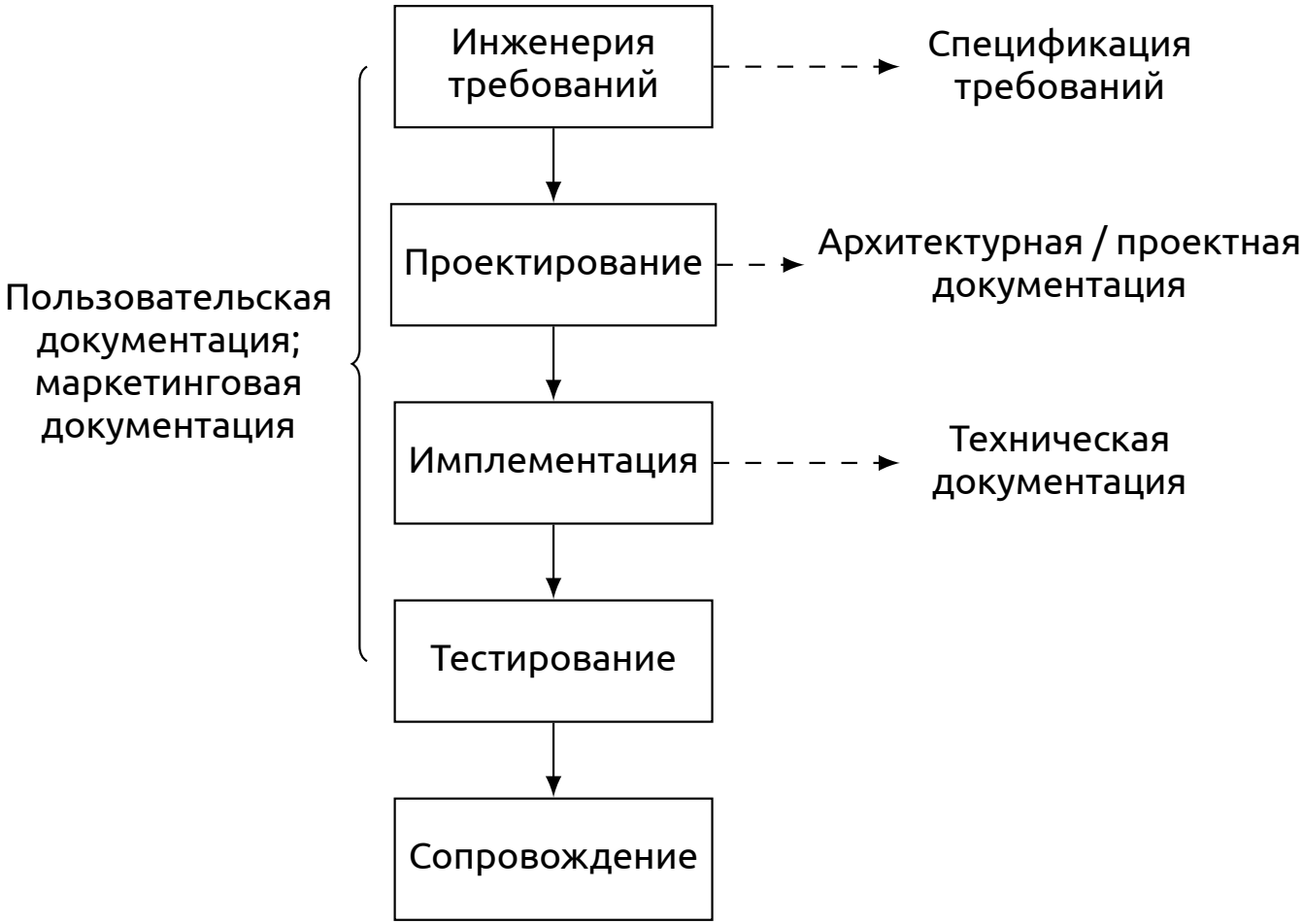
Определение

Документация — печатный текст, сопровождающий программное обеспечение для объяснения принципов его функционирования или использования.

Цели документирования:

- посредничество между разработчиками ПО;
- упрощение сопровождения и эволюции;
- информация для планирования и оценки затрат в процессе разработки;
- инструкции по использованию и управлению программной системой;
- основание для сертификации системы.

Документация на ПО



Документирование в процессе разработки ПО

Типы документации

Документация **на процесс разработки** (англ. *process documentation*):

- планы разработки;
- расписания;
- документы оценки качества процессов разработки;
- организационные и проектные стандарты.

Документация **на продукты разработки** (англ. *product documentation*):

- системная (техническая) документация — описание программной системы с точки зрения разработчика;
- пользовательская документация — описание ПО с точки зрения конечного пользователя.

Документация в гибкой методологии

Работающее ПО > *Исчерпывающая документация*

— Agile Manifesto

Недостатки традиционного подхода к документированию:

- Производство документации и поддержка документов в актуальном состоянии занимает много времени и средств и приводит к замедлению процесса разработки.
- Требования к ПО меняются настолько быстро, что документация устаревает практически сразу после написания.

Необходимые виды документации:

- пользовательская документация;
- обоснование архитектурных решений;
- документация критических систем.

Структура документации

Основной стандарт: IEEE 1063 — Standard for Software User Documentation [2001].

Структура документации на ПО:

1. данные, позволяющие идентифицировать документ (заголовок, дата составления и т. п.);
2. содержание;
3. список иллюстраций и таблиц (опционально);
4. введение — назначение документа и краткое описание содержимого;
5. информация по использованию — советы по эффективному использованию различными группами пользователей (новичками, опытными пользователями, администраторами, …);

Структура документации

Структура документации на ПО (продолжение):

6. концепция ПО — описание вариантов использования программной системы;
7. команды — описание команд, поддерживаемых системой;
8. выдаваемые программой сообщения об ошибках и способы их устранения;
9. словарь используемых в документе специфичных терминов;
10. связанные документы и информационные ресурсы;
11. навигация (особенно для электронных документов);
12. алфавитный указатель по командам;
13. поиск по содержанию (для электронных документов).

Стиль документации

- Предпочтительно использование английского языка, в т. ч. из-за проблем перевода терминов;
- проверка грамматики (присутствует в современных средах разработки);
- короткие и ясные предложения; короткие абзацы (не более 7 предложений).
- четкие определения для используемых терминов;
- нумерованные и ненумерованные списки для перечислений, выделение текста (курсив или полужирное начертание);
- заголовки и подзаголовки для фрагментации информации;
- иллюстрации и таблицы для наглядности.

Форматы документации

- **Печатная документация**;
- **Электронная документация:**
  - локальные файлы (plain text, Markdown, HTML, PDF, …);
  - интегрируемая в общесистемную справочную систему (man, info, …);
  - интегрируемая в среду разработки (напр., исходные Java-файлы или javadoc-архивы при разработке на Java в Eclipse).
- **Онлайн-документация:**
  - поддерживаемая разработчиком (руководство по установке / Getting started / справочные руководства, …);
  - Web 2.0-документация, поддерживаемая пользователями (wiki, блоги, вопросы на stackoverflow, …)

Онлайн-документация

Преимущества:

- доступность для потребителей, актуальность документации;
- гипертекстовая связанность в пределах документации и с другими источниками информации;
- больший объем документов;
- веб 2.0 — возможность комментирования документации, обмена опытом с другими пользователями ПО.

Недостатки:

- усложнение поиска по нечетким запросам;
- ухудшение воспринимаемости текста;
- большой объем малополезной информации.

Документирование процессов разработки

Виды документации:

- планы, оценки затрат и расписания: составляются менеджерами для управления процессом разработки;
- отчеты: использование ресурсов на различных этапах;
- стандарты: ограничения на процесс разработки (специфичные для организации или национальные / международные);
- рабочие документы (working paper): особенности архитектуры системы, стратегии имплементации;
- общение между разработчиками и менеджерами.

Объем документации на процесс разработки

Наблюдение

Большая часть документации на процесс разработки может быть заменена неформальными дискуссиями между разработчиками, менеджерами и заказчиком.

Необходимая документация на процесс разработки:

- явно определенная договором с заказчиком;
- необходимая для сертификации системы;
- расписание тестирования (заменяется автоматическими тестами);
- рабочие документы (могут быть выделены в отдельные статьи).

Пользовательская документация

Определение

Пользовательская документация (англ. *user documentation*) — документы, описывающие использование программной системы конечными пользователями.

Организация пользовательской документации:

- учебные пособия — описание шагов для решения определенных задач с помощью программной системы;
- темы — объединение логически связанных документов в главы / разделы, описывающие определенный аспект ПО;
- справочники — перечень выполняемых системой функций.

Виды пользовательской документации

Вид	Потребители	Содержание
Функциональное описание системы	менеджеры, заказчик	обзор системы, описание отличительных особенностей
Руководство по установке	системные администраторы	описание этапов установки системы
Введение (англ. *getting started*)	пользователи	краткое руководство для начального знакомства с системой
Справочное руководство (англ. *reference manual*)	опытные пользователи	детальное описание функционала ПО

Системная документация

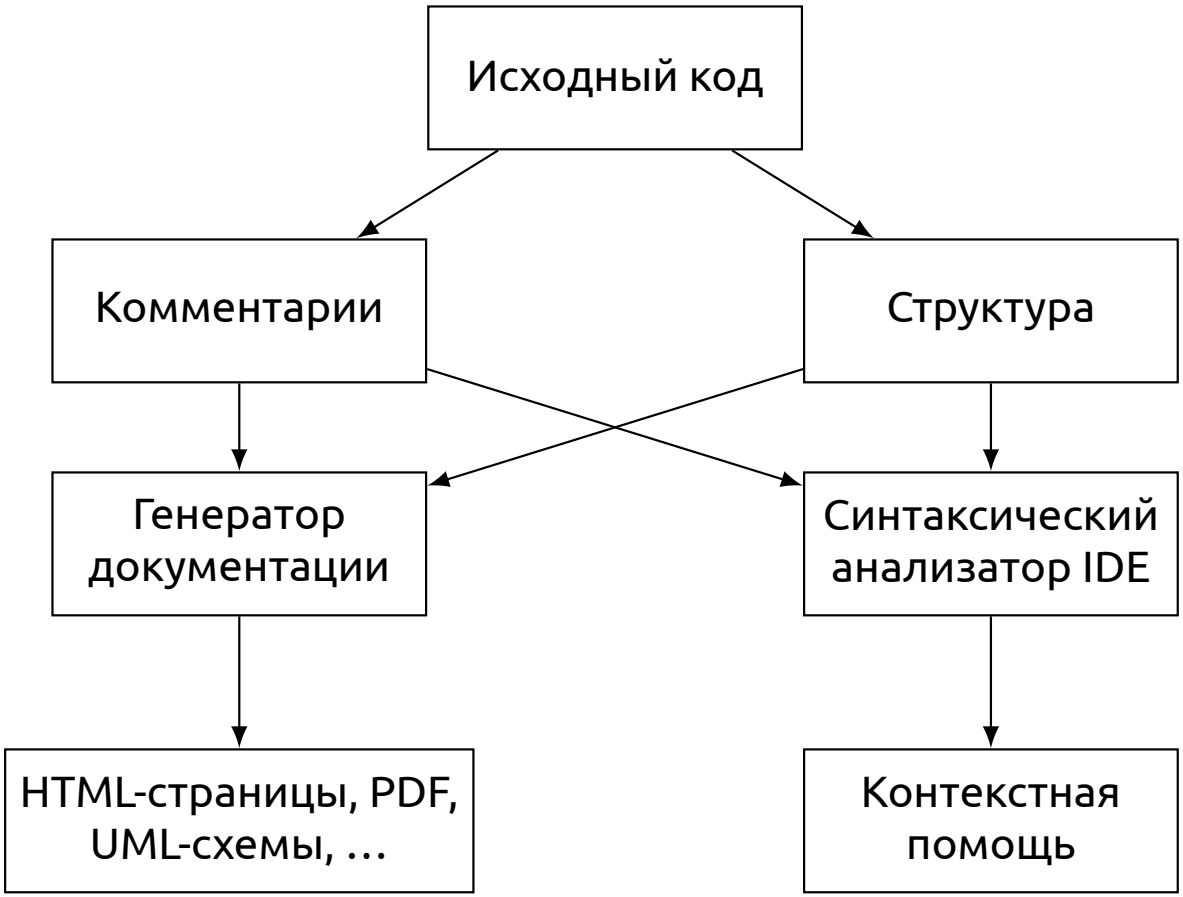
Определение

Системная документация (англ. *system documentation*) — документы, описывающие структуру программной системы.

Виды системной документации:

- документ спецификации требований;
- описание общей архитектуры системы;
- описание отельных компонентов (архитектура, предоставляемая функциональность и интерфейсы);
- исходный код и комментарии в нем;
- документы, касающиеся валидации системы;
- руководство по сопровождению системы (известные проблемы, направления эволюции, внешние зависимости, …).

Генерация документации



Генерация документации на основе исходного кода

Генерация документации

Этапы генерации документации ($\sim$ MVC):

1. определение используемых представлений для исходных файлов;
2. создание синтаксического дерева для исходных файлов;
3. создание моделей для элементов программы (классов, методов, ...);
4. генерация представления на основе моделей (напр., HTML-страниц).

Примеры генераторов документации:

- Javadoc (основной для Java);
- Sphinx (основной для Python);
- Doxygen (основной для C / C++).

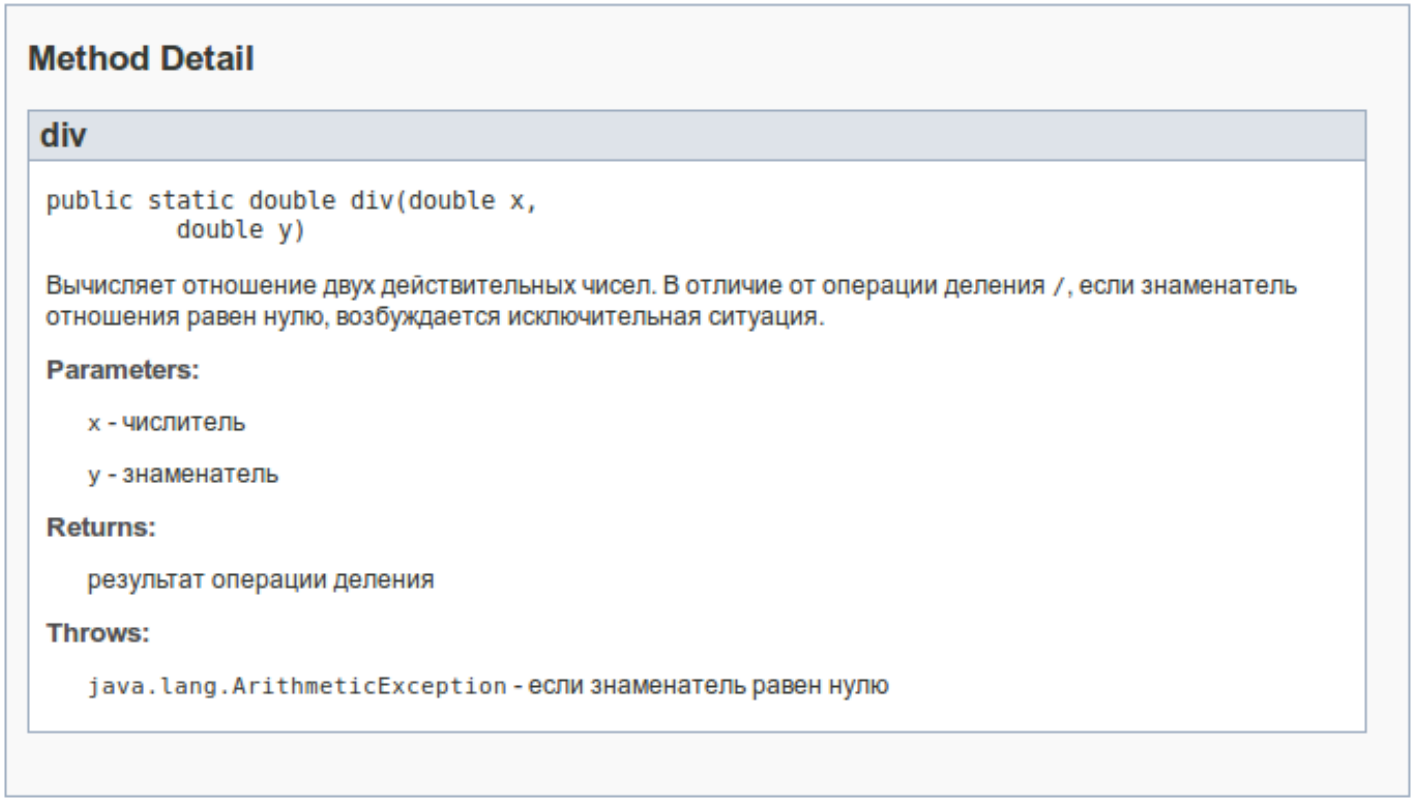
Пример документации: Javadoc

- Для составления документации используются комментарии к классам, полям, методам вида `/** ... */`.
- Для секционирования комментариев применяются теги:
  - `@param` — для описания параметров методов;
  - `@return` — для описания возвращаемого значения метода;
  - `@throws` — для условий порождения исключений;
  - `@since` — для установления версии ПО, в которой появился класс / метод.
- Для маркировки применяются HTML-теги.
- Теги `@link`, `@see` позволяют ссылаться на другие элементы документации.

Пример документации: Javadoc

```
1  /**
2   * Вычисляет отношение двух действительных чисел.
3   * В отличие от операции деления <code>/</code>, если знаменатель отношения равен нулю,
4   * возбуждается исключительная ситуация.
5   *
6   * @param x
7   *     числитель
8   * @param y
9   *     знаменатель
10  *
11  * @return
12  *     результат операции деления
13  *
14  * @throws ArithmeticException
15  *     если знаменатель равен нулю
16  */
17 public static double div(double x, double y) {
18     if (y == 0.0) throw new ArithmeticException("Division by zero");
19     return x / y;
20 }
```

Пример документации: Javadoc



Method Detail

div

```
public static double div(double x,
         double y)
```

Вычисляет отношение двух действительных чисел. В отличие от операции деления `/`, если знаменатель отношения равен нулю, возбуждается исключительная ситуация.

Parameters:

`x` - числитель

`y` - знаменатель

Returns:

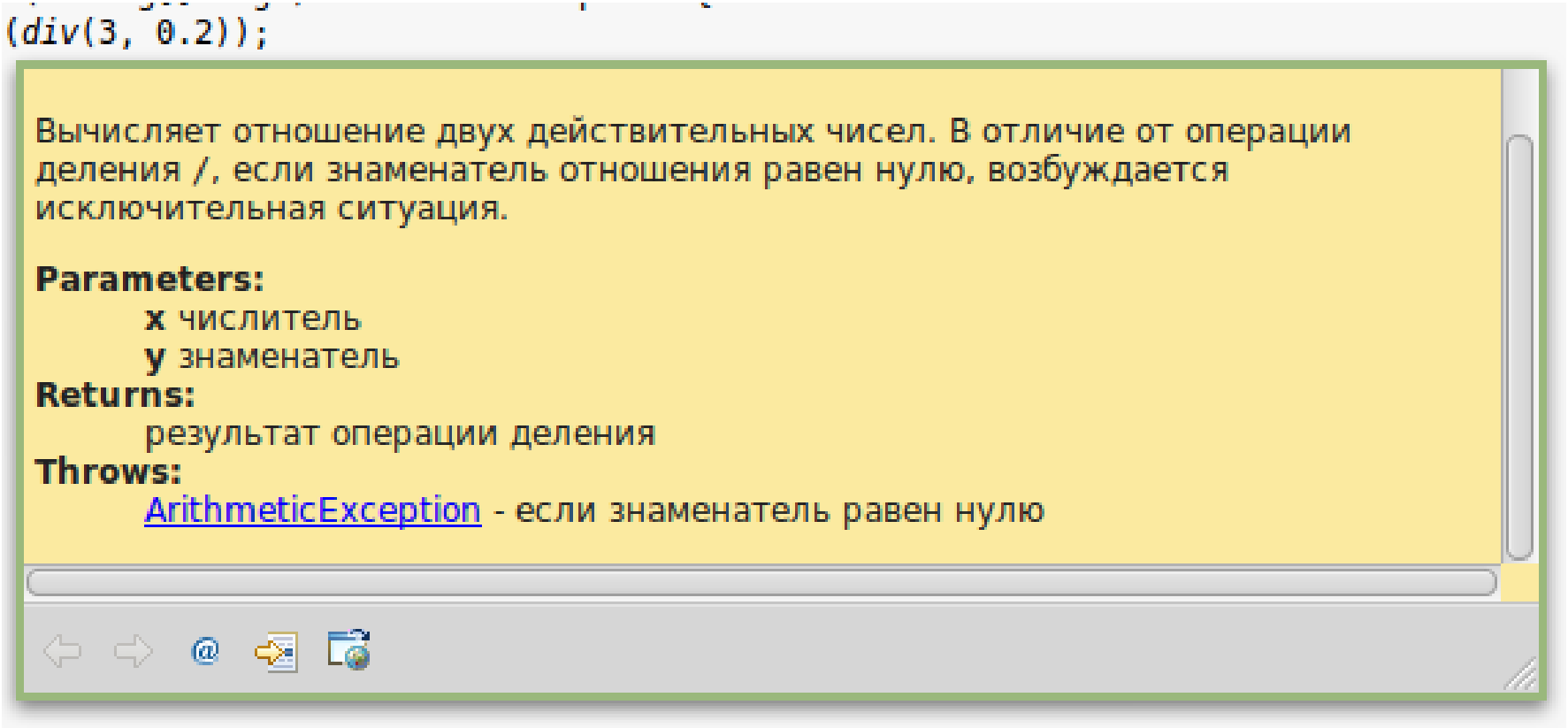
результат операции деления

Throws:

`java.lang.ArithmeticException` - если знаменатель равен нулю

Фрагмент сгенерированной утилитой Javadoc HTML-страницы документации, соответствующей приведенному методу.

Пример документации: Javadoc



Контекстная помощь для приведенного метода в среде разработки Eclipse.

Грамотное программирование

Определение

Грамотное программирование (англ. *literate programming*) — методология программирования и документирования кода, согласно которой программа (*эссе*) состоит из текста на естественном языке с вкраплениями кода на языках программирования, возможно с макроподстановками.

Автор: Дональд Кнут [1981, для системы компьютерной верстки TEX].

Принципы:

- конструирование эссе происходит в порядке разработки программы программистом и следует его мысли;
- роль абстракций выполняют макросы (логически связанные фрагменты кода);
- «чистый» код программы генерируется препроцессором, обрабатывающим макросы; одному эссе может соответствовать несколько программ, в т. ч. на разных ЯП.

Пример грамотного программирования

Пример в системе noweb:

```
1  Our basic C program consists of the five parts:
2  <<*>>=
3  <<Header files to include>>
4  <<Type definitions>>
5  <<Global variables>>
6  <<Function definitions>>
7  <<The main program>>
8  @
9
10 As we intend to present information to the user,
11 we should include the standard I/O library.
12 <<Header files to include>>=
13 #include <stdio.h>
14 @
15
16 So it goes.
```

Выводы

1. Документация на ПО нужна, чтобы описать его функционал для пользователей (пользовательская документация) и упростить сопровождение и эволюцию (системная документация).
2. Согласно гибкой методологии разработки, объем документации (в особенности документов, описывающих процесс разработки) должен быть сведен к минимуму.
3. Стандарт IEEE 1063 содержит рекомендации по составлению документации к ПО, в частности, приблизительную структуру документов.
4. Создание документации можно автоматизировать за счет использования генераторов документации наподобие javadoc.

Материалы

Sommerville, Ian

Software Engineering. Documentation

Доступна по Интернету.

IEEE Society

IEEE 1063 Standard for Software User Documentation

Спасибо за внимание!