Информационная архитектура on-line документации программных комплексов ЦОС

Чичагов А.В.

В работе рассматривается один из подходов к организации процесса разработки и документирования наукоёмких программных продуктов, в частности, программных комплексов цифровой обработки сигналов. Описывается конструкция простого фреймворка пакетов технологической и эксплуатационной on-line документации специализированных программных комплексов ЦОС.

• проблема • проект • процесс • артефакт • документация • база данных • концепция • конструкция • реализация • апробация • программный комплекс • наукоёмкий программный продукт

One approach to software development and documentation of DSP open problem solutions is considered. The simple framework of packages technological and operational online documentation of DSP software systems is described.

• problem • project • process • artifact • documentation • data base • conception • construction • architecture • specification • implementation • approbation • program • software

Введение

Исходной точкой любого проекта является некоторая объективная или субъективная потребность или, иначе, *проблема*, т.е. неформально поставленная задача, которую требуется решить. Итоговой точкой проекта является *продукт*, т.е. материальное и/или интеллектуальное произведение — *результат* решения поставленной задачи.

В юридическом плане *продукт* представляет отчуждаемую интеллектуальную собственность и предназначен для передачи (продажи) и использования внешними пользователями. Это накладывает определённые формальные требования на его содержание и представление. Они могут различаться для разных аудиторий пользователей, что часто приводит к созданию линейки продуктов на базе общего ядра.

В отличие от *продукта* непосредственно *решение* обычно содержит ряд важных информационно-технологических артефактов, которые отсутствуют в теле

продукта. Эти артефакты не играют существенной роли при использовании продукта как изделия, однако, имеют большое значение для развития/модернизации продукта как системы. Информационно-технологические артефакты составляют внутреннюю документацию проекта, которая обычно «закрыта» и не предназначена для передачи внешним заинтересованным лицам.

С другой стороны, для специалистов большинства предметных областей знаний публикация научно-технических статей, содержащих математические модели, вычислительные методы и другие новации в открытых специализированных журналах, является сложившейся практикой. Аналогично, кроме так называемых «закрытых» исследований и разработок, в мире существует довольно много реальных коллективов и виртуальных объединений программистов, которые открывают исходные коды разработанных программ для общего доступа (движение «*open source*»). Выбор открытого или закрытого способа разработки обычно обусловлен не научно-техническими соображениями, а соображениями безопасности и/или финансово-экономическими причинами, которые во многом обязаны архаичному институту защиты интеллектуальной собственности.

Однако если внутреннюю документацию проекта оформить в подходящем виде и обеспечить к ней доступ внешним заинтересованным лицам, то она может стать важным информационным ресурсом, который будет иметь ценность не только для самих разработчиков наукоёмкого программного продукта. Поэтому актуальной задачей является унификация/стандартизация и повышение качества проектной документации, т.е. организация и представление артефактов проекта в виде *продукта* или, иначе, научно-технического произведения.

Технологическая и эксплуатационная on-line документация наукоёмких программных продуктов (НПП), в частности, специализированных программных комплексов цифровой обработки сигналов (ЦОС) включает: исходный код программ, математические модели, вычислительные методы и алгоритмы, по которым строились программы, критерии оценки качества (адекватности) ЦОС, сценарии, исходный код тестов и результаты тестирования модулей (компонент) системы, описание архитектуры/конструкции целевой системы, инструкции пользователя и другие артефакты или информационно-технологические документы.

Пакет указанных артефактов проекта представляет специализированную базу знаний, которую (при доступности для всех заинтересованных лиц) можно назвать *открытым решением (open problem solution)*. В информационно-архитектурном аспекте пакет on-line документации «*открытого решения*» определяется концептуальной схемой (шаблоном) класса пакетов технологической и эксплуатационной on-line документации проекта.

Заметим, что с технической точки зрения понятие «*открытое решение*» является естественным расширением понятия «*открытого программного обеспечения*», однако, в юридическом плане существует ряд вопросов, которые пока не имеют однозначных ответов.

В настоящей работе рассматривается один из возможных подходов к организации процесса разработки наукоёмких программных продуктов, формальной основой которого является некоторый каркас (шаблон) или фреймворк технологической и эксплуатационной on-line документации *открытого решения* задачи.

Организационная модель процесса разработки НПП

Проектом называется ограниченная по времени последовательность согласованных действий одного или группы специалистов, направленная на достижение определённой цели, например, создание наукоёмкого программного продукта (НПП). Список или диаграмма указанных действий с привязкой к временным меткам называется *планом*, а идеальная типовая последовательность действий (без акцента на конкретную цель) — *процессом* или, в рассматриваемом случае, процессом разработки.

Реальной движущей силой любого проекта НПП является команда разработчиков, т.е. *кадры* или, шире, *персонал* проекта. Другим важным элементом проекта является *процесс*, который включает *организацию, административно-финансовую поддержку и управление* проектом. Деятельность команды разработчиков НПП состоит в информационно-семантическом поиске, адаптации/создании, редактировании и документировании цепочек артефактов проекта.

Предполагается также наличие мотивации и достаточно высокой квалификации специалистов и руководителя (администрации) проекта, в противном случае, результатом проекта будет являться, в лучшем случае, некачественный продукт. Общая схема бизнес-модели процесса разработки НПП в виде модели организации («малого предприятия») приведена на *рис.1.*

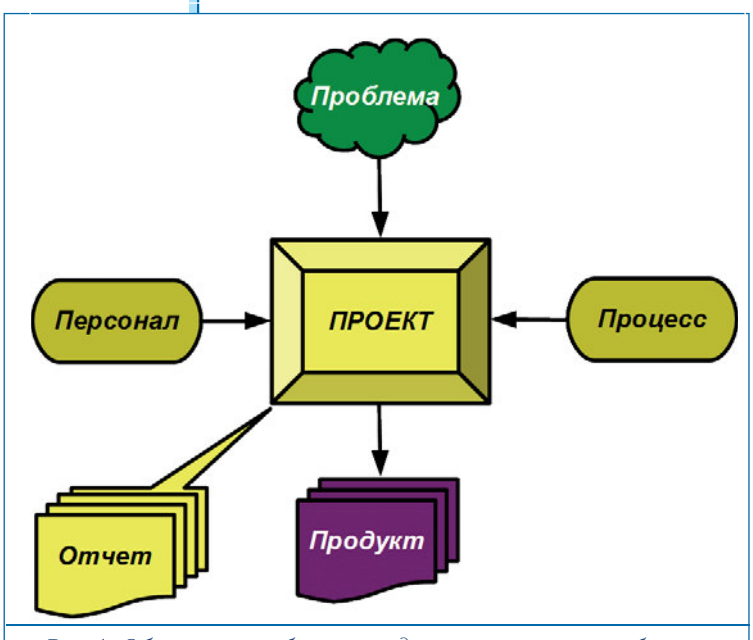

Рис.1. Общая схема бизнес-модели процесса разработки

Важной составляющей процесса является конкретный план работ. Однако при создании НПП, как показывает практика, ни руководитель, ни команда проекта в целом не имеют априори достаточно информации для того, чтобы написать полный, детальный и при этом на 100% достоверный план работ. Поэтому в проектах по созданию НПП принято использовать итерационный подход, включающий регулярное обновление или коррекцию текущего плана работы.

Руководитель проекта, исходя из общей концепции *решения задачи* и текущего состояния проекта, составляет и согласовывает со специалистами рациональный и сбалансированный план работ на текущий временной интервал (итерацию). Интервалы времени, соответствующие итерациям, могут изменяться по ходу проекта, если в этом возникает необходимость.

Проект начинается с того, что разрабатываются общая концепция наукоёмкой программной системы, архитектура и крупнодисперсные спецификации подсистем или компонентов. Затем производится декомпозиция подси-

стем по специализированным функциональным и архитектурным компонентам. После нескольких шагов декомпозиции функциональность компонента получается легко обозримой, и становится возможным выполнить её программную реализацию.

Специализированный функциональный компонент «транслируется» в программный модуль — небольшую программу или небольшой набор программ, которые совместно реализуют простую (элементарную) функциональную абстракцию. Далее, созданные программные модули отлаживаются, тестируются, документируются и интегрируются в разрабатываемую целевую программную систему.

Процесс разработки наукоёмких программных продуктов или, на техническом языке, наукоёмких программных систем (изделий), включает следующие виды работ: анализ (изучение) проблемы и постановка задачи, конструирование (проектирование) системы, реализация (кодирование) и апробация (испытание) изделия. Семантическим идентифицирующим признаком вида или «характера» работы является специализация, т.е. язык, который используется в процессе решения и на котором представляется результат работы. Нужно отметить, что специалист, выполняющий работу, должен владеть, как минимум, двумя специализированными языками — *входным* (чтобы понять задачу) и *выходным* (чтобы описать решение и представить результат).

При разработке специализированных программных комплексов ЦОС работы по разработке системы обычно разделяют на «концептуальный», «логический» и «физический» проекты. «Концептуальный» проект представляет собой описание модели обработки данных и интерпретацию функций системы со стороны предметной области. Иными словами, взгляд на проблему и пути её решения, с точки зрения пользователя. «Логический» проект содержит описание архитектуры системы в виде набора объектов, служб и взаимосвязей между ними. Также рассматривается модель управления обработкой данных, GUI (графический интерфейс пользователя) и модель хранилища данных. «Физический» проект представляет взгляд на систему, с точки зрения программиста, и содержит классовую модель системы, спецификации форматов данных и API (интерфейс программных модулей), а также исходный код программных модулей и модульных тестов.

Общая схема жизненного цикла процесса разработки наукоёмких программных систем приведена на *рис.2.* На этой схеме узел «*Концепция*» представляет результаты анализа (изучения) проблемы и формальную постановку задачи. Довольно часто этот вид работы называют «подготовка технического задания или формулировка требований».

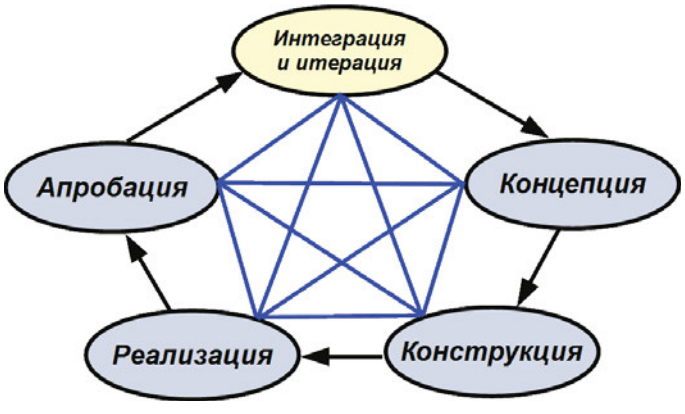

Рис.2. Схема жизненного цикла проекта

Очевидно, что некоторые научно-технические результаты анализа проблемы не всегда легко формально представить в виде «списка требований». Поэтому также используют и другие виды представления информации, такие, например, как информационный обзор,

аналитическая записка и пр., которые могут содержать математические формулы, диаграммы, алгоритмы, варианты использования, ссылки на первоисточники и др. Отметим, что целью документа является простая, понятная и полная передача научно-технической информации между специалистами, а не следование ортодоксальной бюрократической форме.

Узел «*Конструкция*» представляет результаты проектирования системы. «Конструкторская» документация обычно содержит описание архитектуры системы, интерфейса пользователя, спецификации форматов данных и пр.

Узел «*Реализация*» представляет спецификации программного интерфейса и исходные коды модулей целевой программной системы, а также исходные коды модульных тестов.

В проектах, организация процесса в которых соответствует инкрементной модели разработки, кодирование, как и другие виды работ, не является отдельным этапом работы, а распределяется равномерно по всему интервалу времени, которое занимает проект. В таких проектах программные модули кодируются не после того, как будут подготовлены спецификации всех модулей программной системы, а сразу после подготовки спецификации конкретного компонента (небольшой группы модулей) системы.

Результаты кодирования компонентов системы аккумулируются в технологической документации или «базе знаний» проекта, которая включает как исходный код программных модулей системы, так и исходный код модульных тестов. Общую структуру пакета технологической документации программного комплекса ЦОС можно представить в следующем виде:

Структура технологической документации

- Эскизный/концептуальный проект:
  - постановка задачи/техническое задание;
  - аналитическая/математическая модель ЦОС.
- Детальный/логический проект:
  - алгоритмы (процедурные модели компонентов и сценарии тестов);
  - архитектура (объектная модель системы и модель управления ЦОС/GUI).
- Рабочий/физический проект:
  - спецификации форматов данных и интерфейсов модулей (API);
  - реализация (исходный код модулей и модульных тестов).

При создании документации, размер которой выходит за рамки обычных человеческих возможностей по оперативному восприятию содержащейся в ней информации (т.е. приблизительно 5–10 страниц текста формата A4), повышенное внимание необходимо уделять качеству организации документации как системы семантически связанных артефактов.

Наиболее значимыми требованиями к качеству документации являются [1]:

- *Понятность/правильность (understandability/correctness).* Документация должна быть основана на терминологии, принятой в заданной предметной области, и не должна содержать неоднозначных толкований или артефактов, противоречащих друг другу. Каждый термин должен иметь один и тот же смысл во всех документах.

- *Полнота (completeness).* Документация должна содержать всю необходимую информацию для использования приложения специалистами заданной предметной области знаний. В документации должны быть описаны функции системы, доступные элементы интерфейса пользователя приложения, учебные материалы. Документация должна содержать инструкции по конфигурации, управлению и администрированию приложения.

- *Простота обозрения (ease of overview/usability).* Документация должна быть достаточно удобной для обозрения, иметь интуитивно понятные (хорошо известные) средства навигации (оглавление, поиск, предметный указатель и др.). Организация документации должна быть продуманной, гармонической, хорошо сбалансированной и лёгкой для освоения пользователем.

Другими важными требованиями к профессиональной документации считают:

- логическую последовательность и чёткость изложения информации;
- краткость, конкретность, простоту и точность формулировок;
- убедительность, выразительность и содержательность используемой информации.

Разработка документации является скорее искусством, чем наукой, поэтому кроме рассмотренных выше основных требований к качеству документации, при создании документации необходимо опираться на здравый смысл и основные принципы творческой деятельности. В качестве «золотого правила» творческой деятельности приведём известное высказывание А. Эйнштейна: «Всё нужно упрощать до тех пор, пока это возможно, но не более того».

На схеме жизненного цикла проекта *(рис.2)* узел *«Апробация»* представляет эксплуатационную документацию, которая включает контрольно-измерительную, справочно-информационную и учебно-демонстрационную on-line документацию. Контрольно-измерительная документация содержит описание критериев, методов и средств оценки качества цифровой обработки сигнала, а также результаты испытаний (оценки качества) разработанного изделия.

Содержание справочно-информационной документации или, иначе, документации пользователя *(«user's guide»)* хорошо известно и представляет описание возможностей системы, последовательность действий пользователя по установке и конфигурированию системы, а также on-line справку по интерфейсу пользователя *(«help»)*. Учебно-демонстрационная документация содержит учебное пособие *(«tutorial»)* и примеры использования системы *(«samples»)*. Общую структуру пакета эксплуатационной документации системы можно представить в следующем виде:

Структура эксплуатационной документации

- Контрольно-измерительная документация:
  - критерии, методы и средства оценки качества (адекватности);
  - результаты испытаний по оценке качества (адекватности) системы.

- Справочно-информационная документация:
  - руководство пользователя («user's guide»);
  - справка по интерфейсу пользователя системы («help»).

- Учебно-демонстрационная документация:
  - учебное пособие («tutorial»);
  - примеры использования («samples»).

Важной особенностью эксплуатационной документации систем цифровой обработки сигналов является наличие метрологической или контрольно-измерительной документации, в которой представлены результаты проведённых испытаний по оценке качества/адекватности системы.

Целью испытаний является дифференцированная оценка качества компонентов и интегральная оценка качества всей системы. Для объективного сравнения качества различных вариантов изделия необходимо определить или выбрать критерий *«оценка качества»*. Выбранный критерий оценки должен быть простым, понятным и достаточно информативным для рассматриваемого класса систем.

Существуют различные критерии оценки качества. Так, в области цифровой обработки сигналов одним из основных качеств системы является адекватность или оценка степени соответствия реальной цифровой обработки сигнала декларируемой или теоретически ожидаемой. Для широкого класса алгоритмов/программ ЦОС таким критерием оценки качества может быть выбрана оценка частотной кривой адекватности («оценка ЧКА») [2].

Последним из рассматриваемых узлов схемы жизненного цикла процесса разработки наукоёмких программных систем является узел *«Интеграция и итерация»* артефактов проекта. В отличие от рассмотренных выше четырёх узлов, которые соответствуют научно-техническим артефактам, данный узел представляет механизм управления и поддержки проекта.

На схеме *(рис. 2)* этот узел можно лаконично назвать *«Администрация»* (проекта). Члены группы «Администрация» поддерживают (сопровождают) специализированную базу знаний (документов) проекта, т.е. собирают, сохраняют и охраняют от несанкционированного доступа важную проектную информацию, а руководитель проекта контролирует текущее состояние и, с учётом мнений технических специалистов, определяет текущий план работы.

Администрирование проекта состоит в рациональном использовании имеющихся ресурсов для разработки НПП за отведённое время и включает:

- инфраструктуру (материально-техническую базу, финансово-экономическое и юридическое обеспечение, организационные и административные ресурсы);
- управляющий процесс (компетенции, полномочия, стимулы, арбитраж, ответственности участников проекта);
- процесс разработки (методы, инструменты, языки, базы данных/знаний, сопровождение проектной документации);
- текущие планы работ (что, где, когда, кто делает).

Интерактивный подход, который обычно используется в процессе разработки НПП, позволяет:

- рационально задействовать в процессе разработки весь персонал проекта: технических писателей, тестеров, программистов, архитекторов, аналитиков и др.;
- разрабатывать и испытывать наиболее критические компоненты системы в первую очередь;
- изменять/модифицировать разработанные научно-технические артефакты проекта, ограничиваясь «малыми потерями»;
- использовать разработанные компоненты и инструментальные средства при разработке новых программных модулей;
- использовать профессиональный рост навыков сотрудников в процессе развития проекта;
- изменять стратегию и улучшать тактику процесса разработки.

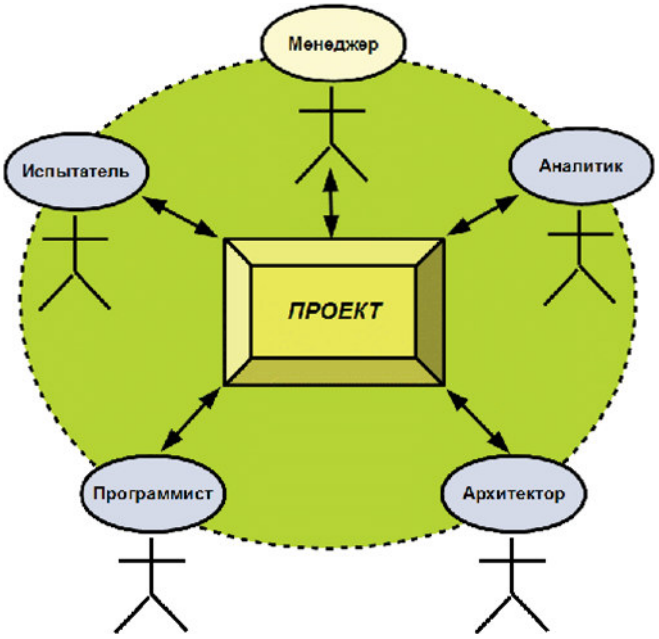

Рис.3. Команда проекта

Итеративный подход предполагает тесное сотрудничество специалистов различных областей знаний в процессе разработки НПП *(рис. 3)*, а следование рациональной методологии и использование стандартов и шаблонов позволяет сэкономить много времени на обдумывании порядка работы и структуры документации проекта. Актуальные стандарты отражают лучшие практики разработки программных систем и напоминают о важных вещах, которые легко упустить.

С другой стороны, каждый проект разработки НПП имеет конкретные особенности, которые не всегда оптимально соответствуют стандартной схеме. Например, для небольших проектов часто нерационально использовать в полном объёме дисциплину разработки «*рациональный унифицированный процесс (RUP)*» [3, 4]. В таких случаях обычно используют дисциплину разработки «*экстремальное программирование (XP)*» [5]. В каждом конкретном случае следует использовать гибкую дисциплину разработки [6], которая рекомендует применять только те практики стандартных процессов разработки, которые соответствуют данному проекту.

Рассмотренная организационная модель процесса разработки НПП определяет жизненный цикл и общую структуру работ. С информационной точки зрения *проект* можно рассматривать как творческий процесс создания проектной документации (отчёта), в котором описывается один из вариантов решения поставленной задачи.

Существующие российские стандарты (ГОСТы), определяющие требования к программной документации, сильно устарели и фактически могут служить лишь в качестве первоначальной информации по данной теме. Поэтому при разработке профессиональной

пользовательской документации следует опираться на международные стандарты, например, IEEE 1063–1987 (R1993) Standard for Software User Documentation (Стандарт на пользовательскую документацию к программному обеспечению) и ISO/IEC 12207 Software Life Cycle Processes (Процессы жизненного цикла программного обеспечения) и др.

Как показывает практика, создание качественной технологической и эксплуатационной документации требует выполнения большого количества итераций. При этом создаваемая документация должна иметь простой и удобный доступ для изучения и редактирования специалистами, типовую структуру и стандартную компоновку.

В информационно-архитектурном аспекте пакет on-line документации определяется концептуальной схемой (шаблоном) класса пакетов технологической и эксплуатационной on-line документации проекта. Ниже описывается конструкция простого, легко расширяемого HTML-фреймворка пакета on-line документации.

Конструкция HTML-фреймворка «SxP-2011»

HTML-фреймворк «*SxP-2011*» предназначен для рациональной организации набора семантически связанных информационных ресурсов (on-line документов) и обеспечения удобного доступа пользователя к этому множеству ресурсов. В структурном плане HTML-фреймворк «*SxP-2011*» представляет контейнер или «пакет» (документов), в котором могут содержаться как первичные информационные ресурсы («рабочие документы»), так и вложенные пакеты документов («суб-пакеты»). В целом структура HTML-фреймворка соответствует топологии дерева, каждый узел которого представляет пакет документов.

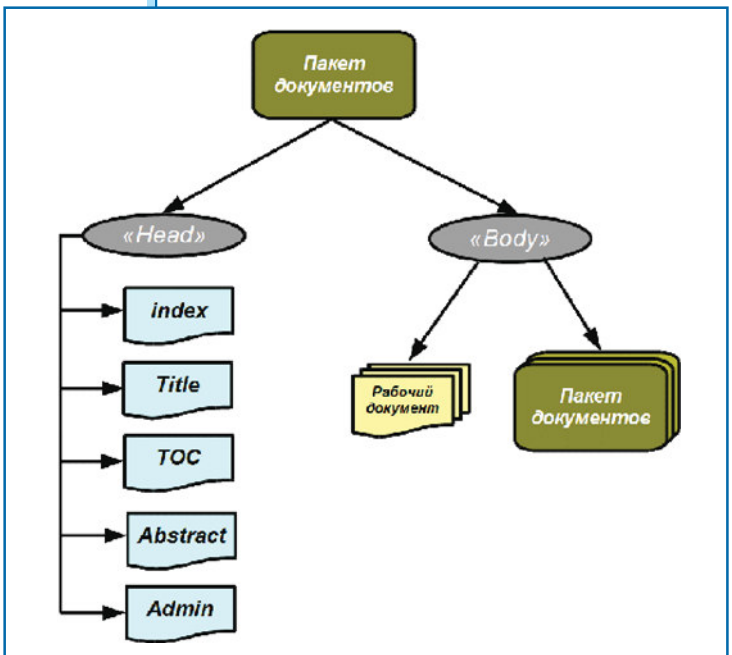

Рис.4. Структура пакета документов HTML-фреймворка «SxP-2011»

Все пакеты документов, построенные на базе HTML-фреймворка «*SxP-2011*» содержат стандартный набор «мета-файлов», назначением которых является унификация визуального представления пакета в Web-броузере и реализация семантической связности набора информационных ресурсов с помощью механизма гипер-ссылок. На логическом уровне пакет можно представить в виде простой структуры, состоящей из двух абстрактных объектов, один из которых можно назвать «*головой (head)*», а другой — «*телом (body)*» пакета *(рис. 4)*,

«Голова» пакета представляет набор мета-файлов, которые отвечают за визуальное представление (компоновку и стиль), навигацию по дереву пакетов и доступ к содержимому «тела» пакета из Web-броузера. «Голова» пакета содержит стандартный набор следующих HTML-файлов:

- *index.html* — основной служебный файл, является точкой входа в пакет, реализует фреймовую структуру пакета документов;
- *Title.html* — верхний колонтитул («шапка») пакета, содержит название и логотип пакета документов;
- *TOC.html* — представляет навигационную панель пакета документов, содержит список ссылок/названий документов пакета (названия или содержательные имена документов часто не совпадают с именами файлов, которые содержат эти документы);
- *Abstract.html* — представляет аннотацию документов, включённых в пакет;
- *Admin.html* — нижний колонтитул, содержит дополнительную информацию пакета документов.

Содержание файлов *«Title.html», «TOC.html», «Abstract.html» и «Admin.html»* определяется автором/составителем пакета и соответствует их именам.

Кроме рассмотренных HTML-файлов, «голова» пакета также содержит специальную папку *«sxp_config»*, в которой содержится набор служебных файлов, определяющий стиль визуального представления и поведение пакета документов (таблицы стилей, js-скрипты, графические ресурсы и пр.). Факультативно рассмотренный набор может включать и другие информационные ресурсы, например, файл, содержащий лицензию и пр. *(рис. 5)*.

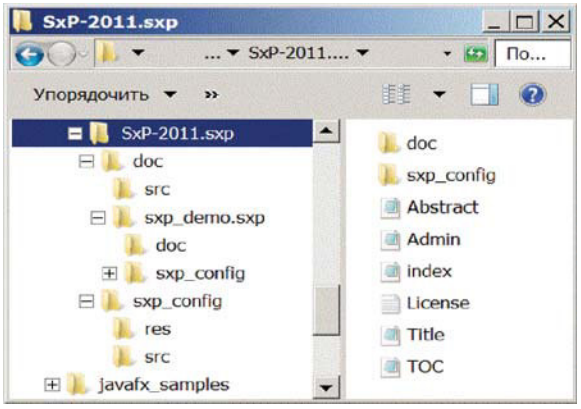

Рис. 5. Реализация пакета документов HTML-фреймворка «SxP-2011»

«Тело» пакета представляет папка «*doc*», которая может содержать как первичные информационные ресурсы («рабочие документы»), так и вложенные пакеты документов. Выделение отдельной папки для хранения документов существенно сокращает время поиска нужных документов в файловом менеджере.

Для быстрого различения пакетов документов от обычных папок рекомендуется выбрать унифицированное имя (аббревиатуру) расширения имени папки, которая представляет пакет документов, например, «*.sxp» (т.е. «simple extensible package» или «simplex package») и использовать в дальнейшем эту аббревиатуру как идентификатор пакета документов (рис.5). Пример презентации пакета документов HTML-фреймворка *«SxP-2011»* показан на *рис. 6*.

HTML-фреймворк *«SxP-2011»* представляет простую систему документационного обеспечения проектов, которая использует Web-браузер и другие внешние приложения для создания/просмотра специальных технических документов. Следует отметить, что рассмотренный HTML-фреймворк *«SxP-2011»* предназначен для использования в «небольших» наукоёмких проектах. В проектах разработки «больших» систем следует использовать более «продвинутые» системы коллективной разработки и документационного обеспечения

Рис.6. Снимок экрана главной страницы пакета документов HTML-фреймворка «SxP-2011»

проектов, такие, например, как «IBM Rational Application Developer» [7] или продукты семейства «IBM Workplace» [8]. Однако гибкие методики разработки [9], на которых основаны эти продукты, можно использовать на любой инфраструктуре, если творчески применять их на практике.

Заключение

Как показывает практика, использование HTML-фреймворка «*SxP-2011*» значительно облегчает авторам работу по созданию «пакетно-структурированной» on-line документации, а читателям обеспечивает простой и удобный способ доступа к документам с помощью Web-браузера.

Литература

1. Чичагов А.В. Разработка интерактивных справочных систем Java-приложений. // Модели и методы распознавания речи. М.: ВЦ РАН им. А.А. Дородницына, 2011. С. 4–45.
2. Чичагов А.В. Оценка адекватности класса спецпроцессоров цифровой обработки сигналов // Программная инженерия. 2011. № 7. С. 37–45.
3. Брауде Э. Технология разработки программного обеспечения. СПб.: Питер, 2004.
4. Кватрани Т., Палистрант Дж. Визуальное моделирование с помощью IBM Rational Software Architect и UML. М.: Кудиц-пресс, 2007.
5. Бек К. Экстремальное программирование: разработка через тестирование. СПб.: Питер, 2003.
6. Амблер С. Гибкие технологии: экстремальное программирование и унифицированный процесс разработки. СПб.: Питер, 2005.
7. Фанг Дж. и др. Введение в IBM Rational Application Developer. М.: Кудиц-пресс, 2006.
8. Себестиан Р., Спенсер Д.У. Стратегия и продукты IBM Workplace. М.: Кудиц-пресс, 2007.
9. Хемраджани А. Гибкая разработка приложений на Java с помощью Spring, Hibernate и Eclipse. М.: ООО «И.Д. Вильямс», 2008.

Сведения об авторах

Чичагов Александр Владимирович,
научный сотрудник вычислительного центра им. А.А. Дородницына РАН, г. Москва