УДК 004.05

МОДЕЛЬ РОЛЕЙ КОМАНДЫ РАЗРАБОТЧИКОВ ПРОГРАММНЫХ СРЕДСТВ НА ОСНОВЕ ГИБКИХ МЕТОДОВ

С.Н. НЕБОРСКИЙ

Белорусский государственный университет информатики и радиоэлектроники
П. Бровки, 6, Минск, 220013, Беларусь

Поступила в редакцию 31 октября 2008

Предлагается модель ролей команды разработчиков программных средств, которая эффективна при разработке на основе гибких методов. Предлагаемая модель основана на таких ролях, как менеджер продукта, менеджер проекта, программист и тестировщик. Определены задачи, функции и связи членов команды. Показан механизм расширения данной модели путем выстраивания вертикальной или горизонтальной иерархии.

Ключевые слова: модель ролей команды разработчиков, гибкие методы разработки.

Введение

Гибкие методы разработки программных средств (ПС) предлагают каркас для реализации проектов программной инженерии. Они направлены на минимизацию риска за счет разработки ПС на основе коротких повторяющихся циклов — итераций. Гибкие методы делают акцент на том, что первостепенная ценность — реализация непосредственно требований к ПС, а все остальные активности (написание проектной документации, поддержание моделей архитектуры системы и т.п.) — второстепенны [1].

Для эффективной разработки ПС необходимо четко определять роли команды разработчиков, устанавливать связи между членами команды, их обязанности и ответственность. Когда с конкретным членом команды связана определенная роль, персональная ответственность этого члена растет [2]. В данной работе предлагается модель ролей команды разработчиков ПС при разработке на основе гибких методов. Но прежде чем привести ее описание, следует рассмотреть существующие модели ролей команд разработчиков.

Наиболее детально проработанной является модель MSF Team Model (Microsoft Solutions Framework Team Model — модель команды каркаса решений Майкрософт). В модели команды MSF не предусмотрено единоначалия — все роли одинаково важны, все члены равноправны. Решения принимаются коллективно, разделяется и ответственность в случае провала проекта. В табл. 1 описаны роли модели команды MSF [3].

Недостатком модели команды MSF является наличие избыточных ролей. Практика показывает, что зачастую задачи обучения пользователей, сопровождения ПС, решение некоторых административных проблем может эффективно осуществляться представителями других ролей. Также вызывает сомнение тот факт, что декларируемое равенство ролей действительно способствует согласованной работе членов команды. Очевидно, в проекте всегда необходим человек, обладающим правом принять решение, которое должно быть выполнено командой вне зависимости от их мнения.

В табл. 2 приведены роли, рекомендуемые моделью команды разработчиков OPEN (Object-Oriented Process, Environment and Notation — объектно-ориентированный процесс, среда и нотация) [4].

Таблица 1. **Кластеры ролей модели команды MSF**

Кластер роли	Описание
Управление программой	Организация процесса разработки (но не руководство разработкой), ведение графика работ, проведение утренних совещаний, обеспечение соответствия стандартам и спецификациям
Управление продуктом	Общение с заказчиком, написание спецификации, разъяснение задач разработчикам
Программирование	Разработка и начальное тестирование продукта
Обучение пользователей	Написание пользовательской документации, обучающих курсов, повышение эффективности работы пользователей
Сопровождение	Установка, сопровождение и техническая поддержка продукта, а также материально-техническое обеспечение работы коллектива
Тестирование	Выявление и устранение недоработок и ошибок

Таблица 2. **Модель ролей команды разработчиков OPEN**

Роль	Описание
Архитектор	Проектирование компонентов ПС и руководство командой
Программист	Реализацией компонентов ПС
Тестировщик	Тестирование ПС
Технический писатель	Разработка документации ПС и моделей сопровождающих процесс

Очевидна несостоятельность данной модели, так как в ней отсутствует роль явного лидера проекта. В то же время возникает вопрос, насколько оправдано выделять роль технического писателя, ведь зачастую создание документации является второстепенной задачей.

Можно также отметить такую модель, как модель ролей команды DSDM (Dynamic System Development Method — динамичный метод разработки систем). В табл. 3 приведены роли данной модели [5].

Таблица 3. **Роли модели команды DSDM**

Уровень роли	Роль	Описание
Роли уровня проекта	Спонсор проекта (чемпион проекта)	распоряжается бюджетом и другими ресурсами проекта
	Разработчик требований	определяет, что все требования выявлены и понятны, контролирует риск разработки ПС с позиции неясных требований
	Руководитель проекта	выполняет управление проектом
	Технический координатор	создает архитектуру ПС, контролирует техническое качество проекта
Роли уровня реализации требования	Ведущий программист	руководит командой программистов и гарантирует, что команда как целое работает эффективно
	Представитель заказчика	предоставляет разработчикам знания предметной области
	Программист	выполняет непосредственно кодирование моделей ПС
	Тестировщик	тестирует ПС, проверяет качество ПС
Прочие роли	Советчик	представитель заказчика, высказывающий свое мнение и дающий советы разработчикам
	Технический писатель	документирует требования, соглашения и договоренности
	Администратор	управляет административной стороной проекта, налаживает связь заказчика и разработчика
	Специальные роли	например, проектировщик баз данных

Недостатком модели ролей DSDM является ее избыточность (очевидно присутствие лишних ролей). Спорным моментом является то, насколько обоснованно вовлекать заказчика в процесс разработки в той степени, как того требует DSDM.

Подобно тому, как модель ролей команды разработчиков DSDM применяется при разработке на основе соответствующего динамичного метода разработки систем, существует и модель ролей команды XP (eXtreme Programming — экстремальное программирование), которая применима при разработке на основе метода XP [2]. Роли этой модели описаны в табл. 4.

Таблица 4. **Модель команды разработчиков XP**

Группа	Роль	Описание
Руководящие роли	Тренер	координирует работу команды, решает ее групповые проблемы, следит за процессом разработки
	Контролер	ведет дневник команды, измеряет ее прогресс, соотносит получаемые показатели прогресса с прогнозируемыми значениями
Роли заказчика	Конечный пользователь	проводит повседневное тестирование ПС как конечный пользователь, связывается с другими пользователями для проведения тестирования, собирает и обрабатывает отзывы
	Представитель заказчика	определяет варианты использования ПС, принимает решения о выпусках и итерациях ПС, предоставляет свой отзыв о ПС, разрабатывает приемочные тесты
	Ответственный за приемочное тестирование	работает с заказчиком для определения приемочных тестов, инструктирует других о приемочных тестах
Кодировщики	Ответственный за модульное тестирование	автоматизирует модульное тестирование, инструктирует других членов команды о разработке тестов модулей
	Ответственный за архитектуру	определяет архитектуру ПС, работает над упрощением архитектуры, ищет участки кода ПС, которые необходимо подвергнуть рефакторингу, контролирует надлежащее проведение рефакторинга
	Ответственный за стандарты кодирования и инструменты	определяет и улучшает стандарты кодирования, изучает инструменты разработки, которые могут помочь команде
	Ответственный за эффективность кодирования и корректность	убеждает команду в пользе парного программирования, осуществляет парную проверку кода, ищет участки кода ПС, где можно улучшить его качество
	Ответственный за интеграцию	настраивает среду интеграции, включая управление исходным кодом
Группа сопровождения	Ответственный за презентации	планирует, организует и проводит демонстрации ПС
	Ответственный за документирование	ведет проектную документацию, документирует процесс разработки, разрабатывает инструкции по установке ПС и руководство пользователя
	Ответственный за установку	обеспечивает автоматическую установку ПС

Модель ролей команды XP применима только при разработке ПС на основе метода экстремального программирования. Однако данный метод обладает такими недостатками, как неуправляемый процесс реализации изменяющихся требований и отсутствие моделей, определяющих архитектуру ПС. При определении подходов к разработке ПС необходимо учитывать это.

Модель ролей команды разработчиков

В данной работе предлагается модель ролей команды разработчиков, главная цель которой — позволить эффективно разрабатывать ПС на основе гибких методов. На рис. 1 предлагаемая модель отображена графически.

Предлагаемая модель не ограничивается только данными ролями. В зависимости от конкретного проекта могут вводиться прочие роли (например, аналитик предметной области, технический писатель). Данная модель предлагает каркас ролей, который применяется независимо от условий проекта.

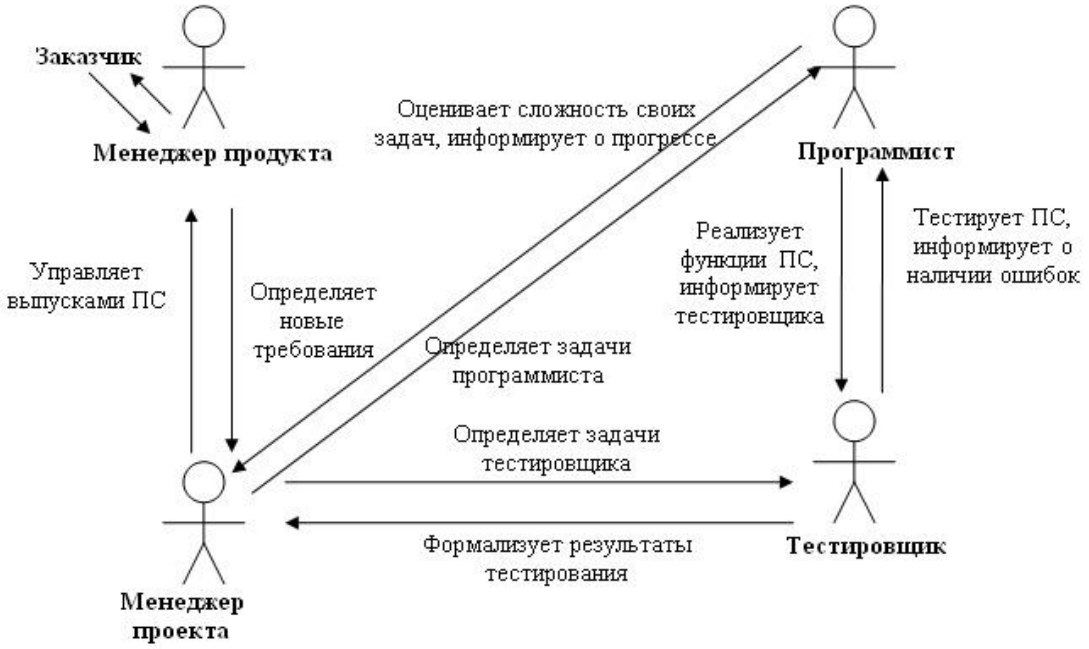

Рис. 1. Взаимодействие команды разработчиков

Схема на рис. 1 показывает типичный сценарий взаимодействия команды разработчиков. Она может быть представлена в виде графа *G(V, E)*. Вершинами его *V* будут являться непосредственно роли команды разработчиков, а дугами *E* — их связи, или взаимодействия. На рис. 2 приведен граф *G(V,E)*, описывающий предлагаемую в данной работе модель.

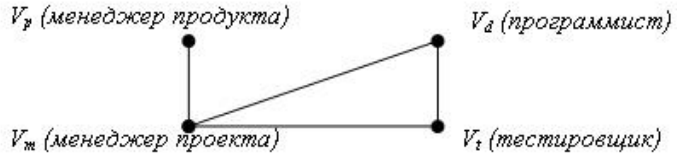

Рис. 2. Модель ролей команды разработчиков

Множество вершин *V* приведенного графа — это непосредственно роли команды разработчиков:

$$V = \{V_p, V_m, V_d, V_t\}.$$

Несколько членов команды могут принадлежать одной роли. В то же время, предлагаемая модель допускает совмещение ролей, т.е. возможна ситуация, когда один и тот же член команды принадлежит нескольким ролям и выполняет соответствующие им функции. Если $P = \{p_i \mid i = \overline{1, N_p}\}$ — множество членов команды мощности N_p, то отображение множества членов команды в множество ролей $F : P \to V$ может быть задано так:

$$F = \begin{pmatrix} p_1 & p_2 & \dots & p_{Np} \\ F(p_1) & F(p_2) & \dots & F(p_{Np}) \end{pmatrix},$$

где $F(p_i) \in V, i = \overline{1, N_p}$

Отображение $F : P \to V$ в общем случае не инъективно, т.е. несколько членов команды могут принадлежать одной роли. Вместе с тем, отображение *F* является сюръективным, т.е. каждый член команды обязательно выполняет некоторую роль. Следует определить задачи и функции членов предлагаемой модели ролей команды.

Задача менеджера продукта — удовлетворение требований заказчика. В его обязанности входит представление интересов заказчика, формирование общей картины решения, формализация требований, принятие решений, касающихся треугольника компромиссов: функции – ресурсы – временные рамки. Как можно заметить, вершины V_p и V_m смежные, причем вершина V_p имеет степень 1: $\deg(V_p) = 1$. Это отражает тот факт, что общение менеджера продукта с командой осуществляется только посредством менеджера проекта.

Целью работы менеджера проекта является поставка ПС, реализованного в заданных условиях. Его область деятельности охватывает координирование коммуникации команды, распределение задач и информирование о статусе проекта, управление проектными рисками, проверка результатов тестирования, управление процессом обеспечения качества, администрирование. Вершина V_m имеет степень 3: $\deg(V_m) = 3$, что свидетельствует о самой высокой коммуникативной нагрузке данной роли.

В обязанности программиста входит оценка времени и сложности каждой задачи, разработка (программирование) ПС согласно спецификации требований, подготовка ПС к поставке, начальное тестирование продукта. Вершина V_d является смежной с вершинами V_m и V_t и имеет степень 2: $\deg(V_d) = 2$.

Область деятельности тестировщика связана с планированием тестирования, тестированием и ведением отчетности по обеспечению качества. В его обязанности входит идентификация проблемных областей ПС, разработка плана и стратегии тестирования, непосредственно проведение тестирования. Вершина V_t является смежной с вершинами V_m и V_d, она имеет степень 2: $\deg(V_t) = 2$. Получается, коммуникативная нагрузка данной роли такая же, как и роли программиста.

Предлагаемые в модели роли — не равноправны. В данной модели выделяется два уровня ролей:

– управляющие роли (менеджер продукта, менеджер проекта);

– исполняющие роли (тестировщик, программист).

Члены управляющих ролей уполномочены принимать решения. Именно они определяют дальнейшую стратегию развития проекта. Безусловно, они несут и большую ответственность, чем представители других ролей.

Расширение модели ролей команды разработчиков

Предлагаемая модель не содержит связи между менеджером продукта и кем бы то ни было, кроме менеджера проекта. Расширение данной модели не может происходить со стороны менеджера продукта. Можно выделить подграф $G'(V', E') \prec G(V, E)$, множество вершин которого включает только менеджера проекта, программиста и тестировщика:

$V' = \{V_m, V_d, V_t\}$

Очевидно, подграф *G'(V',E')* является связным. Это гарантирует эффективность работы команды. Этот подграф интересен тем, что расширение модели команды разработчиков происходит добавлением новой вершины и ребра, соединяющего эту вершину с одной из уже существующих вершин *V'*. При этом добавление новой вершины, смежной с V_d или V_t, означает создание управляющего звена, что в свою очередь говорит о выстраивании вертикальной иерархии. Управляющее звено — это члены новой роли модели команды, которые выполняют управленческие функции по отношению к подчиненным звеньями. Под звеном в данном случае понимается совокупность членов команды, принадлежащих одной роли.

На рис. 3 приведен пример расширения предлагаемой в данной работе модели новыми звеньями программистов.

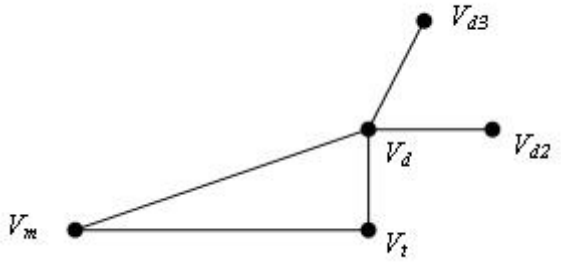

Рис. 3. Расширение модели ролей команды разработчиков новым звеном
(выстраивание вертикальной иерархии)

Для графа на рис. 3 новым управляющим звеном являются члены V_d, которым подчиняются члены ролей V_{d2}, V_{d3}. На практике это может означать, например, существование отдела разработки клиентской части ПС и отдела разработки серверной части ПС, которые подчиняются ведущему программисту, выполняющему интеграцию компонентов ПС.

Сложность данной модели можно оценить исходя из диаметра графа d (для графа на рисунке 3 d=3) и максимальной степени вершин $k = \max\{\deg(V_1), \deg(V_2), ..., \deg(V_N)\}$, $N = |V|$ — количество вершин графа (для графа на рис. 3 k=4). Диаметр d графа определяет оперативность работы команды. Чем больше диаметр, тем больше управляющих звеньев в модели, и, следовательно, тем больше времени требуется на обмен информацией. Степень вершины k определяет ее нагрузку. Чем больше k, тем меньше эффективность работы звена.

Включение в предлагаемую модель новых ролей осуществляется путем добавления новых ребер, смежных с вершиной менеджера проекта V_m. Это означает выстраивание горизонтальной иерархии. На рис. 4 приведен пример добавления новой роли V_n.

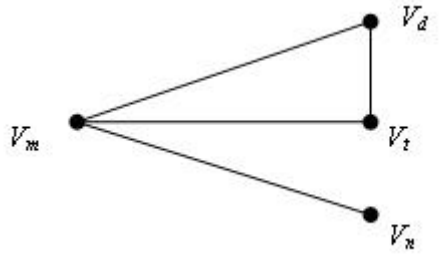

Рис. 4. Добавление новой роли модели ролей команды разработчиков
(выстраивание горизонтальной иерархии)

Для получения оптимальной команды разработчиков, необходимо оптимизировать функцию *f(d,k)* работы разработчиков. В простейшем случае *f(d,k)* можно задать так:

$$f(d,k) = dk.$$

Так как d и k — неотрицательные целые числа, функция *f(d,k)* является возрастающей. Чем больше ее значение, тем менее эффективно работает команда. Следовательно, для выбора оптимальных значений параметров d и k необходимо минимизировать *f(d,k)*:

$$f(d,k) \xrightarrow[d,k]{} \min$$

Заключение

Таким образом, в данной работе предложена и формально описана модель ролей команды разработчиков. Предложенная модель основана на таких ролях, как менеджер продукта, менеджер проекта, программист и тестировщик. Расширение данной модели происходит либо путем добавления нового управляющего звена (выстраивание вертикальной иерархии), либо путем добавления новой роли, связанной с менеджером проекта (выстраивание горизонтальной

иерархии). Применение предложенной модели ролей команды разработчиков на практике должно способствовать повышению качества разрабатываемого ПС, и, в то же время, сокращению времени разработки.

TEAM MODEL IN AGILE SOFTWARE DEVELOPMENT

S.N. NIABORSKI

Abstract

A team model when tackling software project with agile methods is introduced. This model is based on such roles as product manager, project manager, developer and tester. Aims and functions of each team member are defined. An interaction between members is shown. The proposed model can be extended by building vertical and horizontal hierarchy, which is also described in the paper.

Литература

1. *Hunt J.* Agile Software Construction. Springer, 2006.
2. *Dubinsky Y., Hazzan O.* // 5th Int. Conf. on Extreme Programming and Agile Processes in Software Engineering. Garmisch-Partenkirchen, Germany, 2004. P. 57–165.
3. Анализ требований и создание архитектуры решений на основе Microsoft .NET. Учебный курс MCSD. / Пер. с англ. М., 2004.
4. *Firesmith D., Henderson-Sellers B.* The OPEN Process Framework. An Introduction. Addison-Wesley, 2001.
5. *Stapleton J., Constable P.* DSDM: A framework for business centered development. Pearson Education, 2003.