Contents

[Основные роли, встречающиеся в управлении и разработке IT-проекта 1](#_Toc429916155)

[Для чего нужно разделение ролей 1](#_Toc429916156)

[Распределение функций между исполнителями 1](#_Toc429916157)

[Подход компании Microsoft к распределению ролей 3](#_Toc429916158)

[Основные роли, встречающиеся на проекте и их обязанности 4](#_Toc429916159)

[Возможность совмещения ролей 7](#_Toc429916160)

## Основные роли, встречающиеся в управлении и разработке IT-проекта

### Для чего нужно разделение ролей

Каждый IT-специалист идет по пути наименьшего сопротивления, как правило, пытаясь сбросить часть работы на коллегу. Для избегания возможных конфликтов в команде нужно четко разграничить участок работ каждого участника, в том числе заказчика. Для целенаправленного выполнения проекта должен быть выполнен ряд работ, различных как по своему назначению, так и по квалификационным требованиям, предъявляемым к разработчикам. Иными словами, в ходе развития проекта командой разработчиков выполняются те или иные функции.

### Распределение функций между исполнителями

Функции, выполняемые разработчиками*,* — понятие неформализованное. В разных проектах оно может обретать свое содержание. Заметим, что в рамках деятельности менеджера любого проекта необходимо организовать распределение функций проекта между исполнителями. В результате ее выполнения члены команды, выполняющей проект, начинают играть соответствующие роли.  
Обычно роль объединяет родственные функции. Также принято обозначать роли их главными функциями. Продолжая только что приведенную иллюстрацию функций, выполняемых разработчиками проекта, укажем на следующие роли: кодировщик — действующее лицо, главной функцией которого является кодирование, аналитик — тот, кто занимается анализом требований. Подобную характеристику можно дать и тестировщику. Что же касается функции отладки, то в реальных проектах она обычно подразделяется на несколько видов: отладка компонентов, которой занимаются разработчики компонентов (например, те же кодировщики) и комплексная отладка, которая может поручаться специально выделенным сотрудникам или рассматриваться в качестве задачи группы разработчиков. Часто выполняются такие работы, как отладка спецификаций, декомпозиция проекта и др. Иными словами, функция отладки обычно не рассматривается как образующая роль, а распределяется по нескольким ролям в соответствии с принятой стратегией развития проекта. Роли назначаются на начальной стадии жизненного цикла проекта.

Не следует путать функции, которые предписано выполнять разработчику как исполнителю определенной роли, с поручениями в проекте. Поручения — это разовые или систематические задания, из которых обычно и складываются действия, необходимые для выполнения той или иной функции. Если для функции определен регламент выполнения поручений, т. е. последовательность выполнения составляющих ее поручений не требует дополнительных разъяснений для исполнителя, то такая функция называется технологической. В случае нетехнологической функции сотруднику, который выполняет соответствующую роль, приходится самому выстраивать нужную последовательность. Иными словами, технология здесь уступает место ремеслу.

При разработке любых проектов естественно стремление к повышению их технологичности, к установлению регламента для как можно большего числа функций. И одним из способов достижения этого для менеджера является использование работников с нужной квалификацией, для которых поручаемые им роли оказываются технологичными, т.е. состоящими только из технологических функций. К сожалению, реальность такова, что менеджеру приходится работать в условиях ограниченных возможностей в подборе кадров, а потому уровень технологичности выполнения проекта снижается по сравнению с идеальной ситуацией. Таким образом, в рамках любого проекта возникает задача повышения квалификации сотрудников. Для различных схем ведения проектов эта задача решается по-разному. Крайнюю точку зрения на проблему соответствия квалификации работников требованиям проекта отражает идея экстремального программирования, когда используется неформальная организация группы исполнителей проекта без четкого распределения ролей, а значит, и обязательств сотрудников. В этом случае провозглашается принцип, когда каждый в группе делает то, что он умеет делать лучше всего. И хотя все функции, которые должны выполняться, остаются, создается впечатление, что в группе исполнителей проекта исчезают роли. В результате возможны пробелы в разработке, в частности при анализе и декомпозиции проектируемой системы. Чтобы этого избежать, разработчики должны понимать, какую абстрактную роль они исполняют в каждый конкретный момент, выполнение каких проектных функций необходимо сейчас, как связаны между собой работы, как должны распределяться ресурсы. Словом, они должны обращать внимание на выполнение распределенных по группе менеджерских функций. И даже в этом случае схему экстремального программирования можно рекомендовать лишь для слаженных групп исполнителей с высоким уровнем коллективной ответственности.

Функции, выполняемые разработчиками в проекте, подразделяются на:

* Организационные
* Производственные

Первые создают условия для выполнения проектных заданий, вторые непосредственно связаны с этими заданиями. Часто неудачи проекта возникают из-за того, что менеджер не учитывает важность выполнения организационных функций. Так, обычно проектное задание фиксирует лишь то, что нужно предъявлять заказчику в качестве результатов. С точки зрения результатов просто не требуется знать, например, как организована передача проектных материалов между разработчиками, какая процедура отчетности предусматривается, но игнорирование задач реализации информационных потоков в проекте может привести к хаосу, что в конечном итоге отразится и на результатах.

Понятно, что как состав, так и значимость ролей разработчиков и тех, кто с ними связан, различаются в зависимости от выполняемого проекта, от коллектива исполнителей, принятой технологии, от других факторов. Неоднозначность выбора ролей приводит к тому, что их список часто становится непомерно велик. Чрезмерное увеличение числа есть следствие отождествления роли и функции и, соответственно, игнорирования понятия родственности функций. В то же время, если ролей выбрано недостаточно, есть опасность, что не все нужные в проекте функции будут охвачены планированием и управлением.

Также при выборе состава нужно учитывать направление того или иного человека. В случае крупных проектов необходимо нанимать узкопрофильным специалистов, так как они лучше знают свою сферу. Если у проекта небольшой бюджет, то можно сэкономить, наняв многопрофильных специалистов. К слову, есть такой класс разработчиков, как full-stack разработчики — это еще одна попытка «работодателей» получить задешево, то, что никогда дешевым быть не могло. Это такой же психологический прием, как и «вы же профессионал! Вы же профессионал?»

Цепочку распределений ролей можно описать в следующем виде:

* Спонсор (куратор) проекта (это сотрудник (как правило, руководитель высшего звена) организации, реализующей проект, который курирует проект со стороны организации (владельца проекта), обеспечивает общий контроль и поддержку проекта) назначает менеджера проекта и обеспечивает ему необходимую поддержку.
* Менеджер проекта выбирает команду управления проектом, среди которых есть командный лидер (team leader).
* Командный лидер назначает разработчиков.

### Подход компании Microsoft к распределению ролей

Как конкретный разработчик может получать одновременно несколько ролей, так и роль может быть распределена между несколькими исполнителями. Когда менеджеру в конкретных условиях руководства коллективом придется распределять роли, он неизбежно столкнется с тем, что эта задача зависит и от специфики проекта, и от контингента исполнителей. В связи с этим уместно упомянуть об одном из ее решений, которое предлагается специалистами Microsoft в качестве универсального подхода.

Предлагается образовывать небольшие мобильные коллективы как атомарные производственные единицы с общей ответственностью за выполняемые задания — так называемые проектные группы. Такие группы строятся как многопрофильные команды, члены которых распределяют между собой ответственность и дополняют области компетентности друг друга. Группа состоит не более чем из 10 человек. Все они считаются обладающими сходным уровнем профессиональной подготовки, хотя и в разных областях индивидуальной специализации. Провозглашается равноправие членов группы и коллективная ответственность за выполняемые задания: проектная группа — команда равных. Все это позволяет сохранять внутри группы неформальные отношения.  
Вместо понятия роли для группы в целом определяются **ролевые кластеры**, которые заполняются точно так же, как происходит распределение ролей. В то время как за успех проекта ответственна вся команда, каждый из ее ролевых кластеров, определяемых моделью, ассоциирован с одной из проектных целей и работает над ее достижением. В данной модели именно эти цели задают роли разработчиков, которые определяются кластерами. В терминологии используется понятие области компетенции, или области функциональной специализации (functional area), обозначающее ту или иную роль, которую выполняет кластер группы в проекте. Принципиальное отличие распределения исполнителей по ролевым кластерам от распределения ролей заключается лишь в том, что ответственность за это несет не менеджер проекта, а сама группа. Менеджер проекта выдает задания и контролирует их выполнение лишь в целом для группы, не вмешиваясь в ее работу.

Определено шесть **ролевых кластеров**, которые соответствующим образом структурируют проектные функции разработчиков:

* **Управление продуктом (product management)**. Ключевая цель кластера — обеспечивать удовлетворение интересов заказчика. Для ее достижения кластер должен содержать следующие области компетенции:
  + планирование продукта;
  + планирование доходов;
  + представление интересов заказчика;
  + маркетинг.
* **Управление программой (program management)**. Задача — обеспечить реализацию решения в рамках ограничений проекта, что может рассматриваться как удовлетворение требований к бюджету проекта и к его результату. Области компетенции кластера:
  + управление проектом;
  + выработка архитектуры решения;
  + контроль производственного процесса;
  + административные службы.
* **Разработка (development)**. Первостепенной задачей кластера является построение решения в соответствии со спецификацией. Области компетенции кластера:
  + технологическое консультирование;
  + проектирование и осуществление реализации;
  + разработка приложений;
  + разработка инфраструктуры.
* **Тестирование (test)**. Задача кластера — одобрение выпуска продукта только после того, как все дефекты выявлены и устранены. Области компетенции кластера:
  + разработка тестов;
  + отчетность о тестах;
  + планирование тестов.
* **Удовлетворение потребителя (user experience)**. Цель кластера — повышение эффективности использования продукта. Области компетенции кластера:
  + общедоступность (возможности работы для людей с недостатками зрения, слуха и др.);
  + интернационализация (эксплуатация в иноязычных средах);
  + обеспечение технической поддержки;
  + обучение пользователей;
  + удобство эксплуатации (эргономика);
  + графический дизайн.
* **Управление выпуском (release management)**. Задача кластера — беспрепятственное внедрение и сопровождение продукта. Области компетенции кластера:
  + инфраструктура (infrastructure);
  + сопровождение (support);
  + бизнес-процессы (operations);
  + управление выпуском готового продукта (commercial release management).

### Основные роли, встречающиеся на проекте и их обязанности

Существуют следующие роли на IT-проектах:

* **Заказчик (Customer)** — отвечает за:
  + своевременный просмотр спецификаций и других присылаемых документов (с целью утвердить документ, дать комментарии, исправить неточности и т.п.);
  + внесение замечаний, дефектов, пожеланий в багтрекинговую систему;
  + своевременный просмотр каждого выпуска и предоставление комментариев.
* **Планировщик ресурсов (Planner)**:
  + выдвигает и координирует требования к проектам в организации, осуществляющей данную разработку;
  + развивает и направляет план выполнения проекта с точки зрения организации.
* **Менеджер проекта (Project Manager)** — отвечает за:
* проектная документация;
* составление плана проекта;
* согласование сроков;
* анализ возможных рисков;
* участие в подборе и утверждении проектной команды;
* разбивка продукта на компоненты и раздача их исполнителям;
* определение требуемых ресурсов и рабочей среды, их распределение внутри команды;
* постановка рабочего процесса в команде (разработка, тестирование, работа с требованиями);
* определение приоритетности задач;
* организация работы команды вокруг требуемой задачи;
* отслеживание состояния проекта, хода выполнения задач;
* отслеживание должной приоритетности выполнения задач;
* отслеживание нагрузки задачами и прогресса по задачам каждого разработчика;
* отслеживание сроков выполнения задач;
* удерживание команды в рабочем состоянии, мотивация команды;
* создание прозрачной среды общения между всеми участниками процесса;
* отслеживание удовлетворенности проектом со стороны команды;
* решение всевозможных конфликтных ситуаций внутри команды и в связке заказчик-команда;
* общение с заказчиком, управление его ожиданиями;
* предоставление заказчику отчетности о ходе выполнения задач и проекта в целом;
* презентация заказчику готовых решений, демоверсий, прототипов;
* интервьюирование новых членов команды.
* **Руководитель команды (Team Leader)**

Руководитель команды — это нечто среднее между проектным менеджером и квалифицированным разработчиком. Командный лидер обязан перевести бизнес-задачу в понятную техническую для разработчиков и сказать не только то, что нужно сделать, но и зачем это нужно.

На проектах есть две роли: менеджерская — PM, и техническая — System Architect. Командный лидер отчасти выполняет обе роли, но акцент его обязанностей направлен на менеджмент (акцент на техническую часть — это tech lead).

*Под управленческую роль* TL попадают такие обязанности, как:

* менеджмент;
* распределение и делегирование задач:
* всевозможные оценки и составление рабочего графика;
* контроль состояния проекта;
* проведение митингов;
* коммуникации с заказчиком, руководством и всеми членами команды (разработчиками, архитекторами, тестировщиками, менеджерами).

*Под техническую роль* TL попадают**:**

* участие в написании технической документации;
* выбор технологий для проекта;
* разработка архитектуры;
* обзор и анализ кода (code review);
* контроль и наставление молодых разработчиков;
* проведение технических собеседований;
* грамотное вовлечение новых членов команды в рабочий процесс;
* ответственность за техническую часть проекта.
* **Системный аналитик (Technical Leader) —** отвечает за:
* координацию и контроль деятельности по дизайну, архитектуре и кодированию;
* поддержку контроля версий;
* настройку скрипта для авто-билдера и своевременную сборку версий.
* **Архитектор (Architect)** — отвечает за:
  + проектирование архитектуры системы;
  + согласование развития работ, связанных с проектом.

Архитектор — это человек, который решает, как в конечном итоге будет выглядеть информационная система организации в целом и в деталях. Основная цель архитектора в компании заключается в том, чтобы обеспечить решение задач бизнеса при помощи информационных технологий. Причем, он должен не только сформировать решение, но и контролировать правильность его реализации.

Внутри профессии существуют специализации: *функциональная* и *техническая*. В первом случае архитектор в большей степени отвечает за общение с бизнесом и по результатам контактов определяет конструкцию системы, которая нужна заказчику. Во втором ИТ-архитектор в основном общается с разработчиками и конструирует систему изнутри.

Не всякой компании нужен ИТ-архитектор. На небольших предприятиях или там, где информационные проекты не слишком масштабны, функции ИТ-архитектора может выполнять опытный менеджер проекта, разработчик или иной технический специалист в сфере ИТ.

Иметь собственного ИТ-архитектора необходимо, в первую очередь, крупным компаниям с развитой функциональностью унаследованных систем, разветвленной региональной оргструктурой и имеющим согласованные руководством планы развития ИТ.

* **Эксперт предметной области (Domain Expert)** — отвечает за:
  + изучение сферы приложения;
  + поддержку направленности проекта на решение задач данной области.
* **Разработчик (Developer)**:
  + разработку качественного кода;
  + проведение модульного тестирования;
  + поддержку контроля версий;
  + написание пользовательской документации, относящейся к инсталляции и администрированию.

Это широкое понятие, которое может подразделяться на специальные роли (например, разработчик классов). В зависимости от сложности проекта команда может включать различное число разработчиков.

* **Бизнес аналитик (Business Analyst)** отвечает за:
  + выяснение и анализ всех требований заказчика;
  + фиксирование всех требований заказчика (в багтрекинговой системе и в функциональных спецификациях), прослеживание всех изменений в требованиях;
  + написание и поддержка спецификаций.
* **Разработчик информационной поддержки (Information Developer**):
* создает документацию, сопровождающую продукт, когда выпускается версия. Включаемые в нее инсталляционные материалы, равно как ссылочные и учебные, а также материалы помощи предоставляются на бумажных и машинных носителях.
* Для сложных проектов возможно распределение этих задач между несколькими разработчиками информационной поддержки.
* **Специалист по пользовательскому интерфейсу (Human Factors Engineer)**:
* отвечает за удобство применения системы;
* работает с заказчиком, чтобы удостовериться, что пользовательский интерфейс удовлетворяет требованиям.
* **QA менеджер (Quality Assurance manager) —** отвечает за:
* организацию и контроль процесса тестирования в проекте;
* планирование тестирования;
* участие в адаптации процесса разработки под проект, анализ его качества;
* анализ результатов тестирования и качества продукта;
* участие в управлении требованиями;
* участие в настройке багтрекинговой системы, полное прослеживание багов;
* контроль готовности нового выпуска для QA.
* **QA аналитик (QA Analyst)** — отвечает за:
* подготовку тест дизайна;
* написание тест кейс спецификаций;
* проведение тестирования;
* регистрацию багов;
* прослеживание и проверку багов;
* написание документации пользователя.

### Возможность совмещения ролей

|  |  |
| --- | --- |
| **Роли действующих лиц проекта и совмещение ролей** | |
| **Роли** | **Характеристика совмещения ролей** |
| Менеджер и архитектор | Желательно |
| Менеджер и руководитель команды | Противоречиво |
| Руководитель команды и архитектор | Возможно |
| Руководитель команды и проектировщик подсистемы | Нежелательно |
| Менеджер и разработчик | Не допускается |
| Для различных разработчиков | Эффективно с ограничениями(обычное дело) |
| Создание документации (все сотрудники) | Успешно распределяется |
| Специалист по интерфейсу и менеджер | Разумно |
| Эксперт предметной области и менеджер | Зачастую разумно |
| Специалист по интерфейсу и эксперт предметной области | Редко бывает эффективно |
| Эксперт предметной области и разработчик | Бывает полезно |
| Специалист по интерфейсу и разработчик | Часто полезно |
| Библиотекарь и один из разработчиков | Допустимо |
| Тестировщики и другие члены команды | Перекрестно |
| Эксперт предметной области, тестировщик | Оправданно |

### Распределение ролей посредством RACI-матрицы

Модель RACI — средство для выявления активностей и распределения их по ролям и зонам ответственности. Использование матрицы RACI позволяет избежать непонимания в том, кого необходимо привлечь к проекту, а также кто и что должен делать.

RACI — сокращение от основных ролей участников проекта:

* **Responsible (Исполнитель)**: Тот кому назначена эта роль отвечает за выполнение работы и достижение целей проекта. На каждом этапе может быть несколько исполнителей.
* **Accountable (Ответственный)**:  Исполнитель этой роли отвечает за качество и результаты процесса. Обладатель этой роли обеспечивается полномочиями для обратной связи с исполнителями. На каждом этапе может быть только один ответственный.
* **Consulted (Консультант, Эксперт)**: Тот кому назначена эта роль привлекается, как носитель уникальных знаний или информации. Часто в этой роли выступают эксперты в предметной области.
* **Informed (Информируемый)**: Это лицо, которого необходимо держать в курсе о ходе и результатах процесса, чаще всего в одностороннем порядке, т.к. у него нет полномочий напрямую влиять на ход проекта.

Иногда в эту модель добавляются и другие роли, например, S — supported (Оказывающий поддержку).



Для того, чтобы понимать, по какому принципу такая табличка должна рисоваться, а также как ее использовать на практике (в реальных проектах), рекомендуется уделить должное внимание следующему порядку действий при построении матрицы:

* Определяется список необходимых активностей/процессов в поставленной задаче (проекте)
* Определяется и указывается функциональные роли (людей, которые заинтересованы или которых тем или иным образом касается данная задача)
* Собирается митинг и назначаются RACI коды (собственно — буквы) конкретным ролям, непосредственно разграничиваются ответственности
* Определяются несоответствия (например, слишком много ответственных либо отсутствие таковых)
* Описывается таблица и собираются отзывы
* Контролируется выполнение назначенных ролей

По функциональным ролям, анализировать можем, отвечая на такие вопросы:

* *Много «А»* — правильно ли распределены обязанности? Есть ли в наличии «узкие места»?
* *Много «R»* — не многовато ли ответственности для одной роли?
* *Отсутствие пустых ячеек в таблице* – действительно ли эта роль должна быть вовлечена в такое количество задач?

Также, проводится анализ по выполняемым активностям:

* *Более одного «А»* — только одна роль должна быть подотчетной
* *Отсутствие «А»* — необходимо найти подотчетного
* *Более одного «R» или отсутствие такового* – кто-то должен быть ответственный, однако нетребуется, чтобы ответственность была широко распределена – есть риск того, что задача не будет выполнена
* *Много «С»* — стоит ли консультироваться с многими ролями и будет ли это эффективно?
* *Отсутствие «С» и «I»* — правильно ли установлены коммуникации?

#### Пример использования RACI-матрицы

Допустим, есть авиакомпания, которая на своем сайте собирается внедрить систему online check-in. Глобальные активности, необходимы к выполнению в контексте задачи, будут приблизительно следующие: сбор требований к системе; дизайн решения; непосредственная разработка решения (development); внедрение; собственно – стадия “production”; оптимизация решения.  
  
Далее — определяется список функциональных ролей, в данной задачи возможны такие варианты: внутренний сервис провайдер (IT отдел авиакомпании) или же внешний сервис провайдер в случае отсутствия первого; ISP – компания предоставляющая хостинг для сайта авиакомпании; бизнес подразделение авиакомпании (представляющее интересы заказчика); финансовое подразделение (бухгалтерия); сервис менеджер (в зависимости от размеров организации, может входить во внутренний IT отдел); команда разработчиков (в зависимости от размеров организации, может входить во внутренний IT отдел).

Попробуем расставить RACI коды соответственно ролям и выполняемым ими активностям (ясно, что данный процесс проходит при участии всех сторон).

