УДК 65.015

# ВЫБОР МЕТОДОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ НА ОСНОВАНИИ АНАЛИЗА ПРОЕКТА

***Ж.Д. ДАРМИЛОВА, доктор экономических наук, профессор, профессор кафедры мировой экономики и менеджмента, Кубанский государственный университет***
***e-mail: darmil@mail.ru***

***Д.Ю. САЛЬНИКОВ, магистрант программы «Управление проектами», направление «Менеджмент», Кубанский государственный университет***
***e-mail: d.salnikov23@gmail.com***

## Аннотация

В статье представлен и описан алгоритм выбора методологии, максимально удовлетворяющей требованиям конкретного проекта. Материал подготовлен на основе комплексного анализа ряда проектов по разработке программного обеспечения, реализованных в соответствии с наиболее распространенными методологиями.

***Ключевые слова:*** *методология, алгоритм, управление проектами, итеративная модель, каскадная модель, гибкая методология.*

Сегодня, когда всепроникающий характер информационных технологий способствует повышению эффективности практически во всех экономических секторах, производственно-хозяйственная деятельность не обходится без использования информации в том или ином виде (лицензии, патенты, ноу-хау, книги, знания людей, научные данные и т.д.). Информация, являющаяся носителем знания, превратилась в ресурс или фактор производства, более того, информационные ресурсы являются самым большим потенциальным источником богатства в стране. Основные сферы деятельности в области производства информации в России включают услуги в сфере информатизации, производство программных средств, производство информационных продуктов и баз данных на различных материальных носителях, услуги в области телекоммуникационных технологий, производство аппаратных средств. Одним из важнейших сегментов информатизации в этой сфере является производство программных средств, включающее базовое программное обеспечение и прикладное программное обеспечение. Следует отметить, что технологии разработки программного обеспечения в последние годы стремительно развиваются, тем не менее доля успешных проектов остается невысокой. По данным Минкомсвязи России, в рейтинге регионов по уровню развития информационного общества Краснодарский край занимает лишь 55-е место. Всё это, безусловно, актуализирует проблему информатизации и развития технологии разработки программного обеспечения.

Качество реализации проектов разработки программного обеспечения в значительной степени зависит от используемой методологии. В современных российских реалиях в основе применения методологии обычно лежат стандарты компании, а не стремление максимально упростить процесс разработки и снизить стоимость конкретного проекта. Между тем выбор и применение методологии, максимально удовлетворяющей условиям проекта, может облегчить жизнь, как исполнителю, так и заказчику. Особую актуальность данный вопрос приобретает в сфере разработки программного обеспечения и других высокотехнологичных инновационных продуктов, где требуется быть максимально гибким к различным изменениям.

Однако выбор базовой методологии – это лишь первый шаг. Её следует адаптировать под специфику конкретного и цели проекта, а также исходя из размера и состава команды. Имея полное и сформированное представление о проекте, менеджер может дополнить базовую методологию требуемыми инструментами и техниками из других практик, тем самым повысив производительность проектной группы и оптимизировав сам рабочий процесс.

Сформировать у менеджера IT-проектов понимание процесса выбора базовой мето-

дологии исходя из стоящих перед ним задач – лучший способ оптимизировать процесс разработки и снизить расходы на проект. На рисунке сформирована блок-схема алгоритма подбора методологии, охватывающая большинство основных критериев, влияющих на процесс выбора. Данная схема основана на практическом опыте применения наиболее распространённых методологий разработки программного обеспечения.

Прежде чем перейти к описанию алгоритма выбора методологии, необходимо сформулировать определения ключевых используемых понятий:

− методология – это детализированный набор правил, принципов и практик, составляющих способ реализации той или иной модели. иными словами, методология – это реализация стандарта;

− модель жизненного цикла проекта представляет собой общее описание процесса его разработки;

− фреймворк процессов в свою очередь, представляет собой разновидность методоло-

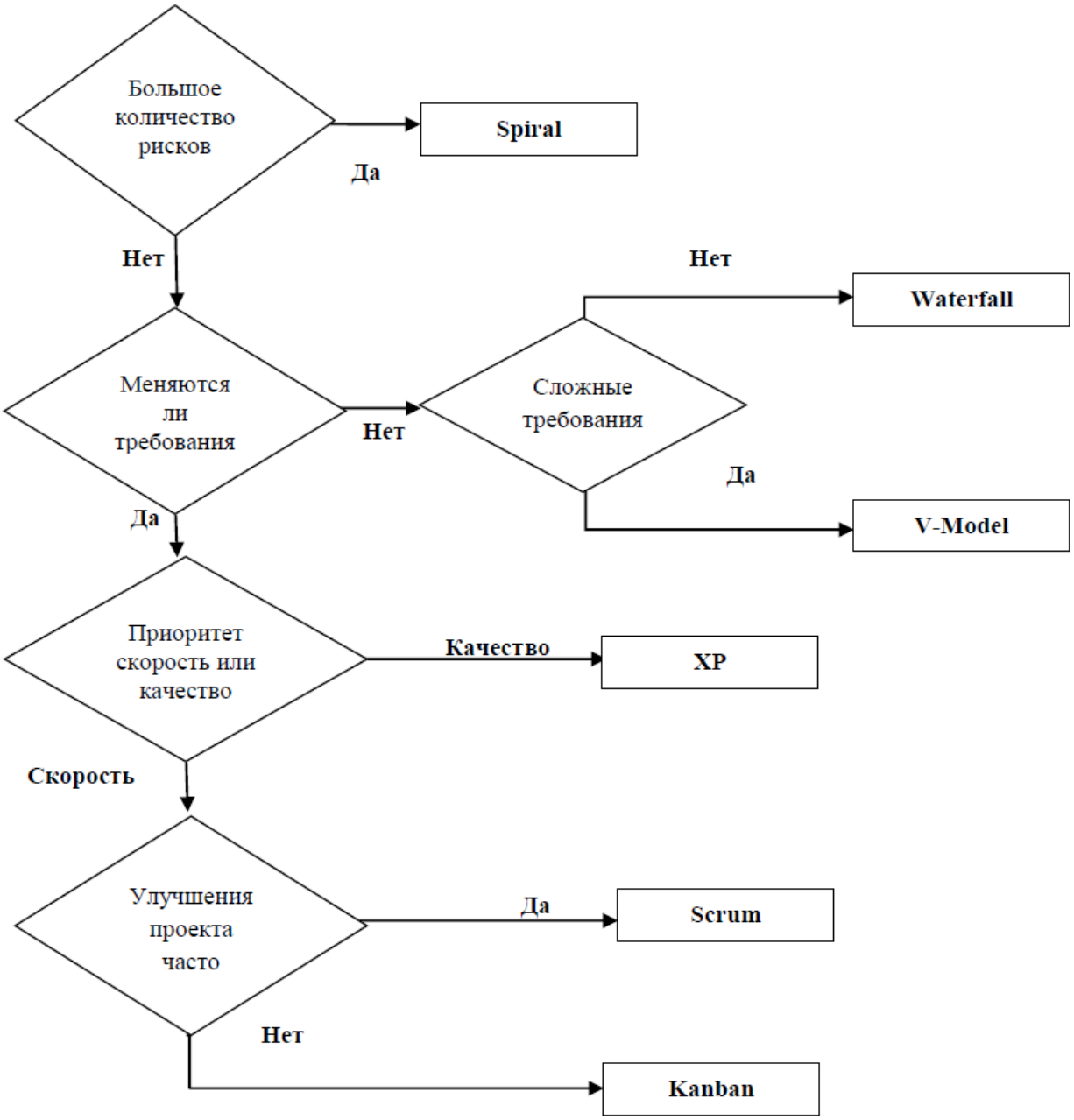


Рис. 1. Блок-схема алгоритма подбора методологии для разработки ПО

гии, содержащую большое количество правил и рекомендаций. При этом пользователь вправе самостоятельно определять и применять лишь те, что подходят под конкретный процесс разработки.

Выбирая методологию управления проектом, стоит начать с первичного критерия – оценки рисков. Риски присущи каждому проекту и опытный проектный менеджер редко берёт в работу заведомо провальный. В случае, если речь идёт о рисках, ставящих саму реализацию проекта под вопрос, оптимально использование *спиральной модели (Spiral)*. Разработка в данной модели представляется в виде спирали. Каждый виток спирали состоит из одной итерации, включающей планирование, анализ рисков, разработку и оценку заказчиком. По итогам итерации стороны решают, является ли продолжение проекта целесообразным. Типичной сферой применения спиральной модели являются исследовательские проекты.

В случае если оценка рисков дала благоприятный результат, следует перейти к следующему этапу – определить возможные изменения в требованиях. Если требования к проекту гарантированно не изменятся, следует остановиться на каскадной модели разработки. *Каскадная модель (Waterfall)* характеризуется чёткой и строгой последовательностью этапов разработки: анализ, проектирование, реализация и тестирование. С её помощью можно оперативно создать продукт без дополнительных накладных расходов на организацию процесса разработки. Главная уязвимость данной модели – неподготовленность к смене требований, так как на этапе подготовки проектируется и описывается весь продукт целиком. Данную модель следует использовать при условии короткого цикла разработки. Однако отсутствие критических рисков, ставящих под вопрос сам факт реализации проекта, не означает, что при работе над ним не требуется вести работы по управлению рисками. Также следует учесть, что необходимость регулярного согласования крупных блоков работы с клиентом зачастую приводит к существенному увеличению сроков разработки и образованию простоев в работе.

В рамках длительного проекта всегда присутствуют технические риски, поэтому каскадная модель будет плохо работать, даже если полностью исключен риск изменения требований, например, если допущена ошибка на этапе выбора инструментов и технологий, разработана некорректная архитектура, недооценена или переоценена требуемая производительность и т. д. Скорее всего, вы узнаете об этом уже на этапе тестирования и релиза продукта, и времени исправить ошибки не останется. Данный момент особенно критичен при работе над стартап-проектами, для которых смена условий и задач зачастую является вопросом «жизни и смерти».

Следующий вопрос – сложность требований. Продумав процедуру тестирования продукта на этапе анализа и проектирования, можно выявить потенциальные проблемы заранее и лучше проработать требования и реализовать архитектуру. В таком случае рекомендуется применение модели *V-Model,* представляющей собой разновидность каскадной модели. Ключевое отличие состоит в том, что этап анализа и проектирования сочетается с этапом тестирования, что помогает лучше сформулировать требования и спроектировать систему. К сожалению, смена требований в ходе разработки характерна и для V-Model. Следует учесть, что её применение в проекте с простыми требованиями гарантированно приведет к существенному росту издержек на разработку.

Если изменение требований неизбежно, следует использовать так называемые гибкие (Agile) методологии, основанные на итеративной модели. Итеративная модель ориентирована на проекты с изменяющимися по ходу разработки условиями. Проект состоит из итераций сроком от 1 до 6 недель (чаще всего 2 недели), включающих этапы анализа, проектирования, реализации, тестирования.

При использовании гибкой методологии следует определиться с приоритетами – скорость или качество. Скорее даже речь идёт о более сложной формулировке: инженерия или продуктивность? Под инженерией подразумевается высокий уровень организованности процесса, использование новаторского подхода и сложных приёмов, применяемых опытной командой. Под продуктивностью – прежде всего возможность оперативного добавления функционала в приложение [2].

В первом случае актуальной становится *методология экстремального программирования (XP)*, состоящая из 12 практик: фиксированная 40-часовая рабочая неделя, стандарты кодирования, простая архитектура, разработка через тестирование, парное программирование, частые релизы, рефакторинг, коллективное владение кодом, непрерывная интеграция, заказчик в команде, игра в планирование, метафора системы. Все 12 практик являются обязательными. Исключение любой

из них приведет к сбоям и по остальным [1]. Помимо этого успешное применение XP требует от команды наличия опыта работы в рамках данной методологии.

В случае упора на максимальную продуктивность есть возможность выбора между *Scrum* и *Kanban*.

*Методология Scrum* ориентирована на постоянное усовершенствование процесса. Проектная команда включает владельца продукта (представитель заказчика), Scrum-мастера (контролирует следование процессу) и остальных членов группы разработки. Итерация, именуемая спринтом, начинается с планирования (митинга), в ходе которого команда определяет и распределяет задачи на итерацию и формирует бэклог спринта. В течение каждого дня проводятся 15-минутные Scrum-встречи (Scrum meeting). По окончании итерации проводятся демонстрация продукта и обзор спринта, где обсуждается, что было сделано хорошо, что плохо и что необходимо улучшить. Также проводится ретроспектива спринта, где обсуждаются идеи и способы улучшения продукта. Оценка поступающих задач, как правило, производится в Стори поинтах (Story points) во избежание ложного чувства контроля над ситуацией, присущего оценке задач в часах. Для этого обычно используются числа Фибоначчи, а участники проектной группы самостоятельно определяют минимальное и максимальные числовые значения, используемые в работе. По истечении первых двух спринтов вычисляется производительность команды (Team velocity), также исчисляемая в стори поинтах [3]. Исходя из данного показателя, при планировании каждого спринта определяется объем задач, которые команда берет в работу в его рамках.

Если на момент старта проекта вы обладаете неслаженной командой либо применяете незнакомые технологии или слабо знаете сферу применения вашего приложения, то выбор методологии Scrum может быть прекрасным решением. В то же время он может отнимать слишком много времени, если у вас отсутствуют перечисленные проблемы. Например, если спринт длится 1 месяц, то обзор спринта по Scrum составляет 4 ч, а ретроспектива – дополнительные 3 ч. Прибавьте к этому 8 ч на планирование спринта и ежедневные пятнадцатиминутные встречи. При этом стоит понимать, что стандартная роль менеджера проекта отсутствует в данной методологии, и вам придётся определиться, какую функцию вы будете выполнять в рамках проектной группы. В зависимости от объема и глубины знаний методологии или же самого продукта менеджер проекта может играть либо роль Scrum-мастера, выполняющего функцию надзора за соблюдением принципов и следования инструментам данной методологии, либо роль Product Owner (владелец продукта) [5].

Для оптимального применения Scrum следует ограничить размеры проектной группы 5–7 специалистами. При этом стоит учесть, что последующее увеличение числа людей, задействованных в проекте, приведет к существенному снижению производительности работы ввиду увеличения сроков коммуникации и зависимостей каждого из них от результатов работы коллег.

*Методология Канбан (Kanban)* представляет собой конвейер задач и базируется на 3 правилах: визуализация процесса разработки с помощью Канбан-доски, постоянное измерение производительности команды и улучшения, а также фиксированный лимит задач в рамках каждого этапа. Он не подходит для ситуации, где требуется разработка сложной архитектуры, так как ориентирован на быстрое добавление функционала, который решает какую-либо задачу пользователю и виден ему. Например, новая страница, отчёт, дополнительные фильтры в поиске. Зачастую работа в рамках Канбан ведется без ограничения рамок спринта (итерации), а бэклог проекта может содержать десятки и сотни задач по поддержке. При этом ключевым правилом Канбана является обязательная оценка критичности каждой из поступающих задач для определения очередности их выполнения [4].

При разработке любого проекта допускается смена методологии по итогам окончания крупного этапа работ. Например, вы можете запустить проект по Scrum, а когда процесс станет более отлаженным и отпадёт потребность в постоянных улучшениях, перейти на Kanban. Семейство гибких методологий находится в постоянном процессе развития и на рынке регулярно появляются как новые подходы, так и вариации старых, например, крайне популярный в последние годы Scrumban. Как следует из названия, она объединяет в себе ключевые составляющие Scrum и Kanban. Однако далеко не все методологии приводят к качественным результатам, которые обещают их создатели и идеологи.

Для обеспечения высокого качества производимого программного обеспечения или иного информационного продукта следует реализовать на предприятии систему контроля качества (Quality Assurance). В неё входит

как ручное и автоматизированное тестирование результатов работы, так и проведение Code review, т.е. анализ качества программного кода и оценка его логичности и чистоты. Внедрение практики непрерывной интеграции (Continuous Integration), обеспечивающей непрерывную выгрузку результатов работы и запуск автоматизированных тестов, позволяет сделать процесс разработки максимально гибким и оперативным.

Выбор конкретной методологии и её адаптация под специфические условия проекта – оптимальный способ оптимизации расходов и производственных издержек компании. Данный перечень практик не является исчерпывающим и включает лишь наиболее распространённые и популярные методологии, применяемые в сфере информационных технологий, как в России, так и за рубежом. Помимо этого также стоит помнить, что грамотная имплементация различных техник и инструментов позволяет существенно оптимизировать базовую методологию и добиться улучшения экономического результата деятельности компании.

**Библиографический список**

1. *Ауэр К., Миллер Р.* Экстремальное программирование. Постановка процессов: пер. с англ. СПб., 2004.

2. *Вольфсон Б.* Гибкие методологии разработки. СПб., 2014.

3. *Кон М.* Scrum: гибкая разработка ПО: пер. с англ. М., 2011.

4. *Расмуссон Д.* Гибкое управление IT-проектами. Руководство для настоящих самураев. СПб., 2012.

5. *Сазерленд Д.* Scrum: революционный метод управления проектами. М., 2015.