.УДК 658.511 DOI: 10.31799/2007-5687-2020-4-45-53

МЕТОДОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ПРОДУКТА

М. О. Гидроец, Л. И. Гришанова
Санкт-Петербургский государственный университет аэрокосмического приборостроения

В данной статье рассматриваются основные методологии разработки программных продуктов. Дана характеристика самым распространенным каскадным и гибким методологиям. Рассмотрены плюсы и минусы каждой из них. Определены ситуации целесообразного применения каскадных и гибких подходов разработки программного продукта.

Ключевые слова: методологии разработки, каскадная модель, гибкая модель, управление проектом, планирование, разработка программного продукта.

Для цитирования:
Гидроец М. О., Гришанова Л. И. Методологии разработки программного продукта // Системный анализ и логистика: журнал.: выпуск №4(26), ISSN 2077-5687. – СПб.: ГУАП., 2020 – с. 45-53. РИНЦ.

SOFTWARE DEVELOPMENT METHODOLOGY

M. O. Gidroets, L. I. Grishanova
Saint-Petersburg State University of Aerospace Instrumentation

This article discusses the main methodologies for developing software products. The most common cascading and flexible methodologies are characterized. Considered the pros and cons of each methodology. The situations of expedient application of cascade and flexible approaches to software product development are determined.

Key words: development methodologies, waterfall model, agile model, project management, planning, software product development.

For citation:
Gidroets M. O., Grishanova L. I. Analysis of passenger air transportation changes under the external environment influence // System analysis and logistics.: №4(26), ISSN 2077-5687. – Russia, Saint-Petersburg.: SUAI., 2020 – p. 45-53.

Введение

Информационные технологии – самая динамично развивающаяся отрасль на сегодняшний день. Современные программные продукты должны быстро подстраиваться под быстро меняющиеся мировые тренды и оставаться конкурентными и рентабельными в агрессивной рыночной экономике. По прогнозам объем российского рынка информационных технологий к 2030 году вырастет в 2,7 раза по сравнению с 2011 годом и составит 4102,6 млрд. руб. [1]. Несмотря на оптимистические цифры роста отрасли огромное количество компаний и стартапов проваливаются, так и не выпустив в свет свой продукт. По статистике только 4 из 10 проектов будут представлены своей целевой аудитории, и только приблизительно 1 из 10 окажется коммерчески удачным. Это связанно со сложной спецификой продукта, имеющей научно–технологическую природу. Стоит отметить и сложный процесс потребления программных продуктов, требующий технической поддержки. Зачастую после того, как продукт попал в руки пользователя, он начинает «жить отдельной жизнью» и развиваться, исходя из потребностей и задач пользователя, что еще более осложняет его поддержку.

Реалии в области информационных технологий на сегодняшний день таковы, что разработку по–настоящему больших программных продуктов могут позволить себе лишь мировые гиганты IT-сферы, а за небольшие проекты, удовлетворяющие сиюминутные тренды, берутся мелкий компании. Но и тех, и других объединяет одна проблема: выбор методологии разработки программного продукта, которая позволит минимизировать затраты на разработку и повысить качество готового продукта [2].

Каскадные методологии разработки программного продукта

Каскадные методологии разработки ПО являются самыми старыми и легкими в планировании методологиями. Весь их принцип заключается в поэтапном создании программного продукта шаг за шагом в соответствии с обозначенной и чётко сформулированной заранее задачей. Каскадные методологии разработки являются наиболее «неповоротливыми» с точки зрения внесения изменений на стадии реализации проекта, четкая поэтапность разработки и необходимость стопроцентной завершенности предыдущего шага для начала последующего не позволяют быстро подстраивать проект под реалии рынка. Реально эффективными данные методологии оказываются при разработке продукта с четко известной целью и функционалом. Такие продукты, как правило, необходимы для удовлетворения корпоративных задач, где цель разработки программного обеспечения (ПО) может не меняться годами, а иногда и десятилетиям. В таких реалиях каскадный метод разработки с поэтапным и скрупулезным выполнением каждого шага позволяет разработать максимально стабильный программный продукт при минимальных затратах ресурсов.

Модель водопада (рис. 1) – одна из самых старых и простых методологий разработки программного продукта. В данной модели наиболее легко управлять проектом. Благодаря жестким рамкам разработки сроки, функционал и стоимость проекта заранее определены. Главным минусом данной методологии является отсутствие возможности сделать шаг назад, т.к. тестирование готового продукта начинается строго после полного завершения разработки. Для внесения каких-либо изменений необходимо ждать полного завершения проекта, что влечет за собой огромные финансовые и временные затраты [3].

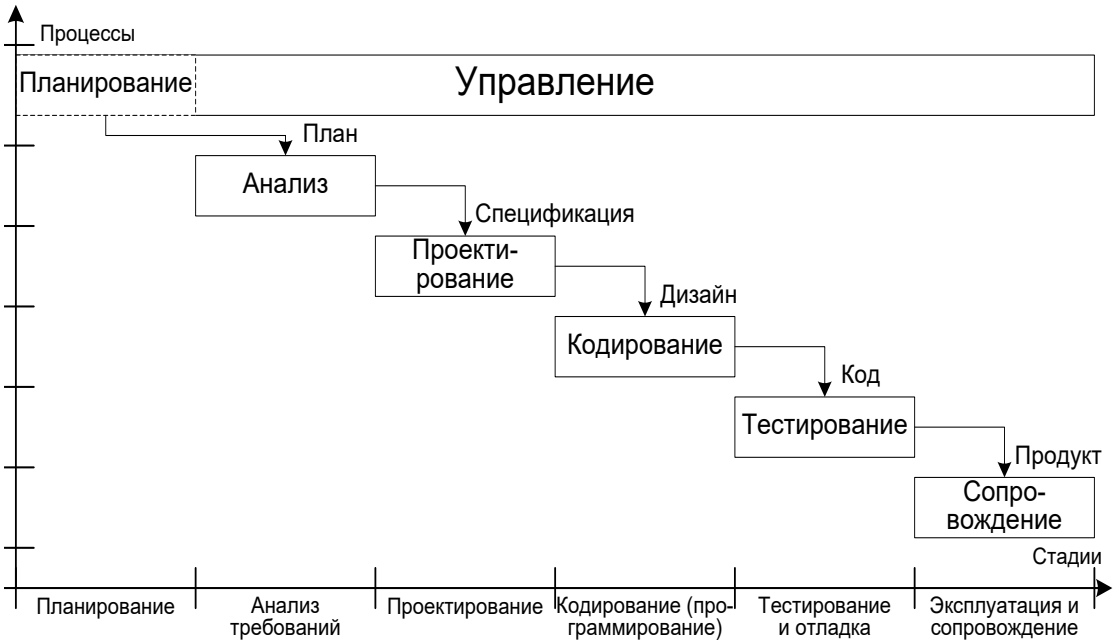

Рис. 1. Модель водопада для разработки программного продукта

V–образная модель (рис. 2) унаследовала структуру «шаг за шагом» от модели водопада. V–образная модель применима к системам, которым особенно важно бесперебойное функционирование. Главной особенностью данной методологии является поэтапная разработка, совмещенная с тестированием, что позволяет получить на выходе стабильно работающий программный продукт. Данная методология является одной из самых ресурсозатратных, но зачастую это оправданно в силу высокостабильного готового продукта. Наиболее активно данная методология используется при разработке программного продукта, где возможность ошибки должна быть сведена к минимуму: системы безопасности, банковские системы, прикладные программы в клиниках.

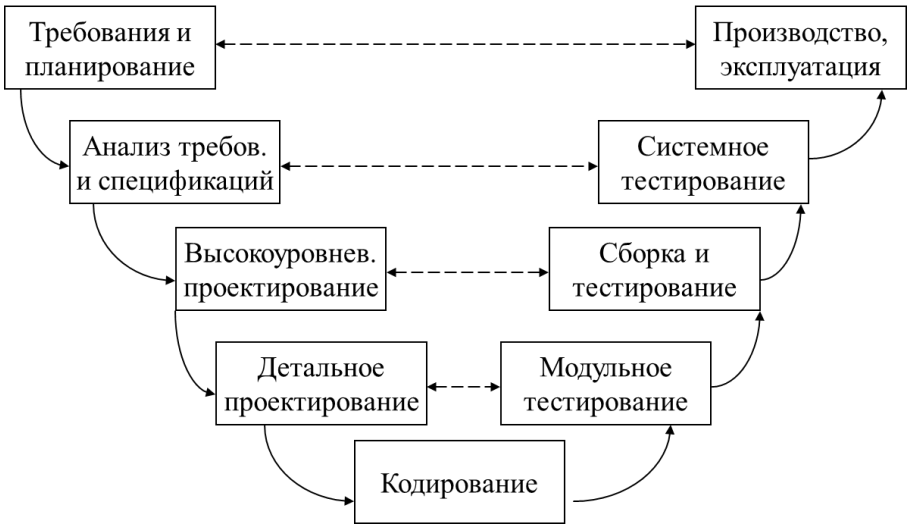

Рис. 2. V–образная модель разработки программного продукта

Инкрементная модель (рис. 3) является более прогрессивной и удобной разновидностью разработки программного продукта. Данная модель вобрала в себя плюсы двух предыдущих моделей, а также позволила сократить затраты ресурсов на разработку продукта. Главной идеей данного метода является поэтапное создание программного продукта при помощи версий (сборок). Процедура разработки по инкрементной модели предполагает выпуск на первом большом этапе продукта в базовой функциональности, а затем уже последовательное добавление новых функций, так называемых «инкрементов». Процесс продолжается до тех пор, пока не будет создана полная система. Большим плюсом данного метода является возможность использовать часть функционала, не дожидаясь полного релиза.

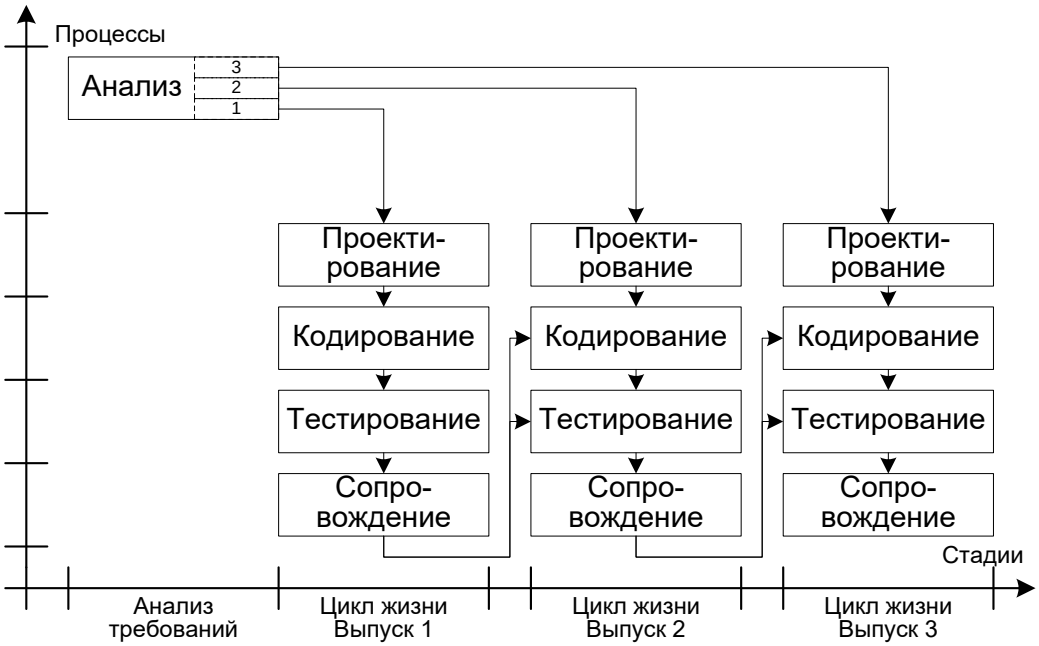

Рис. 3. Инкрементная модель разработки программного продукта

«RAD Model» (rapid application development) или быстрая разработка приложений (рис. 4) является разновидностью инкрементной модели. Суть данного метода заключается в параллельном проектировании компонентов или функций продукта несколькими высококвалифицированными

командами. Временные рамки разработки каждого метода или функции жестко нормированы. Созданные разными командами модули впоследствии собираются в один рабочий прототип, что позволяет предоставить клиенту продукт на обозрение в рекордно короткие сроки. Взаимодействие пользователей с прототипами повышает функциональность проекта, уже на ранних стадиях позволяет выявить главные факторы риска и приспособиться к ним.

Методология RAD подразумевает под собой обеспечение трех главных качеств: высокую скорость разработки, высокое качество конечно продукта и низкую стоимость. Высокая скорость разработки достигается максимально частым созданием прототипов, над которым работают разные команды. Постоянное взаимодействие заказчика и разработчика гарантирует развитие приложения в нужном русле, интерфейс будет удобным и понятным пользователю, а функциональность востребованной, что, в свою очередь, приближает качество продукта к эталонному. Возможность, дающая пользователю самому решать, какой функционал ему требуется в первую очередь, постоянный релиз новых прототипов – то есть, фактически, работающих версией программы – страхуют заказчика от неудач. Если финансирование внезапно иссякнет, пользователь не останется «у разбитого корыта», а благодаря быстрой разработке программный продукт намного раньше выходит в свет, что, в свою очередь, и обеспечивает экономию финансов.

К минусам данной методологии можно отнести:

- сложность управления разными командами,
- необходимость детального проектирования (в методологии IDEF до 5го уровня),
- необходимость собирать части программы воедино,
- зависимость от вовлеченности заказчика,
- необходимость большого количества высококвалифицированных специалистов.

Исходя из вышеизложенного можно заключить, что RAD целесообразней всего применять при реализации небольших проектов с ограниченным бюджетом, разработка которых должна проходить в сжатые сроки. Для масштабных систем с высокими требованиями по контролю и планированию лучше выбрать другие методологии [4].

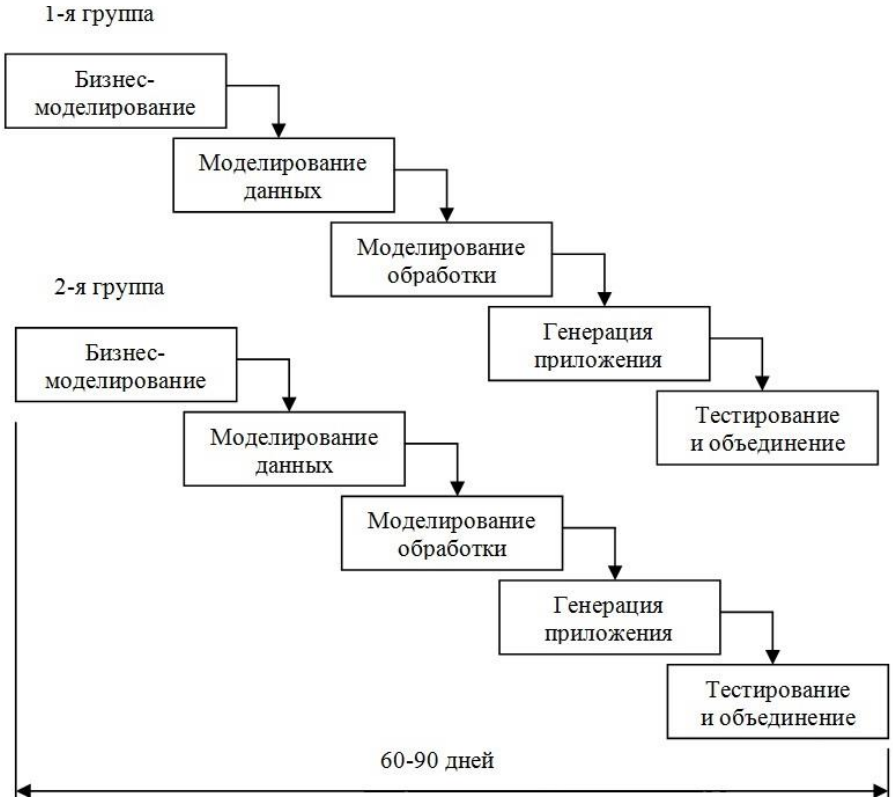

Рис. 4. Модель быстрой разработки программного продукта

Гибкие методологии разработки программного продукта

Появление и быстрое развитие гибких методологий разработки программного продукта связанно с современным развитием экономики. В современных экономических реалиях многие программные продукты должны быстро подстраиваться и адаптироваться к быстро меняющимся трендам и сиюминутным потребностям рынка. Использование в таком случае каскадных моделей разработки просто неуместно, поскольку запланированный продукт может потерять актуальность еще на стадии разработки. Применение гибких методологий оправданно при разработке программного продукта в условиях неопределенности. Эти методологии позволяют изменять план работы и конечный вид продукта практически ежедневно, подстраиваясь под факторы внешней среды: финансирование, спрос, давление конкурентов, потребности рынка, временные рамки. Неоспоримым плюсом гибких методологий также является возможность получения обратной связи от заказчиков или потребителей продукта еще на стадии его разработки, а в некоторых случаях потребители могут принять непосредственное участие в разработке. Главным минусом гибких методологий является сложность управлением проектом, постоянное изменение планов; частая организация собраний не позволяет сосредоточиться рабочей команде на своих целях, в следствии чего происходит падение производительности труда [5].

«Agile Model» (рис. 5) является самой часто используемой моделью гибкой разработки. Суть заключается в быстрых «спринтах», в результате которых появляется некий облик готового продукта. После каждого «спринта» функционал демонстрируется заказчику. В случае удовлетворения потребностей продукт выходит в свет. При возникновении недочетов или новых идей по улучшению продукта проект отправляют на новый «спринт», где он снова проходит все стадии разработки.

Немаловажным отличием данной методологии от каскадных заключается в общности команды. Участники проекта находятся в постоянном контакте друг с другом, в результате чего каждый член команды имеет исчерпывающее представление о предполагаемом виде целевого продукта и проделанной работе коллег. Это позволяет работникам максимально точно приблизить результат своей работы к целевым задачам проекта. Данная методология подходит для больших или нацеленных на длительный жизненный цикл проектов, постоянно адаптируемых к условиям рынка.

Главным недостатком Agile является сложность оценивания трудозатрат и стоимости проекта. Постоянное получение обратной связи и устранение ошибок приводит к постоянным переносам дедлайнов, тем самым создавая угрозу бесконечно продолжающейся работы. Вследствие отсутствия конкретно сформулированных задач приходится постоянно изменять и подстраивать документацию проекта [6].

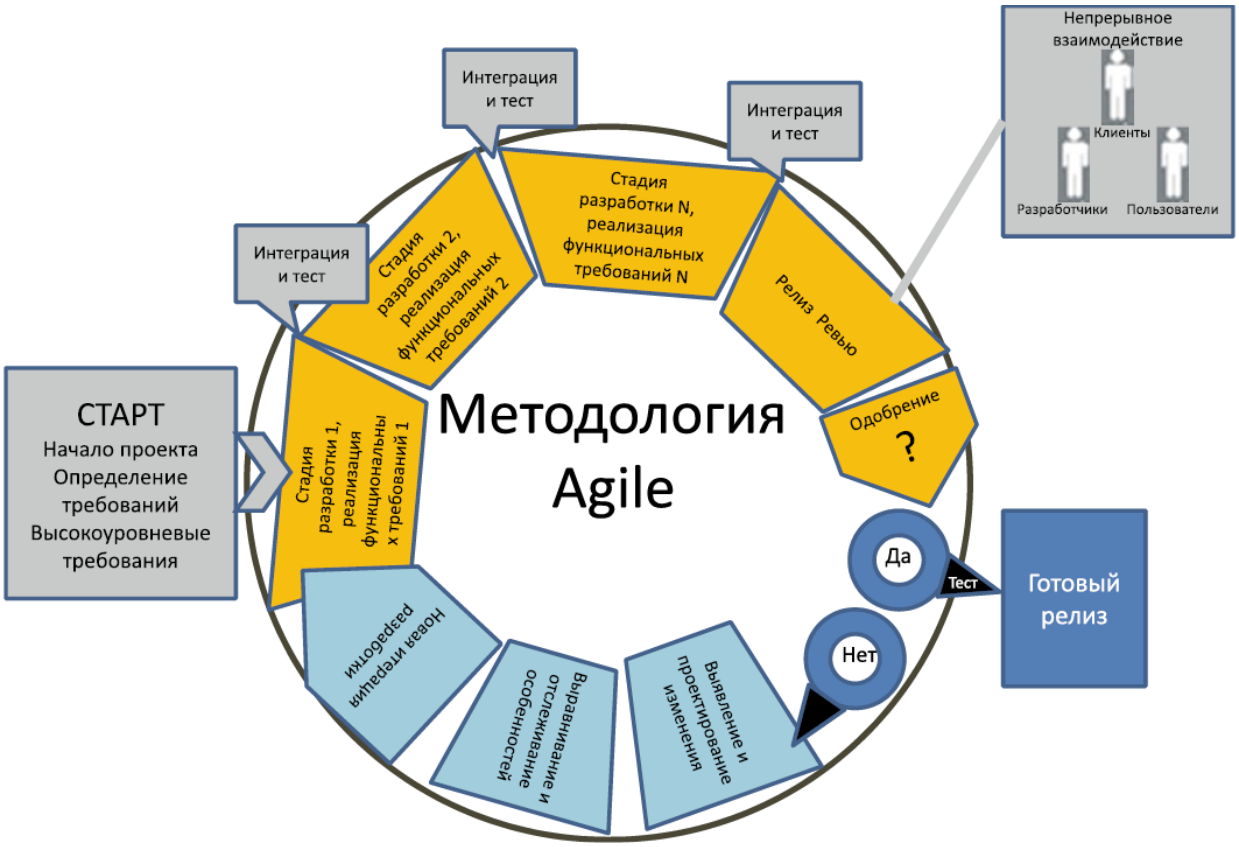

Рис. 5. «Agile Model» разработки программного продукта

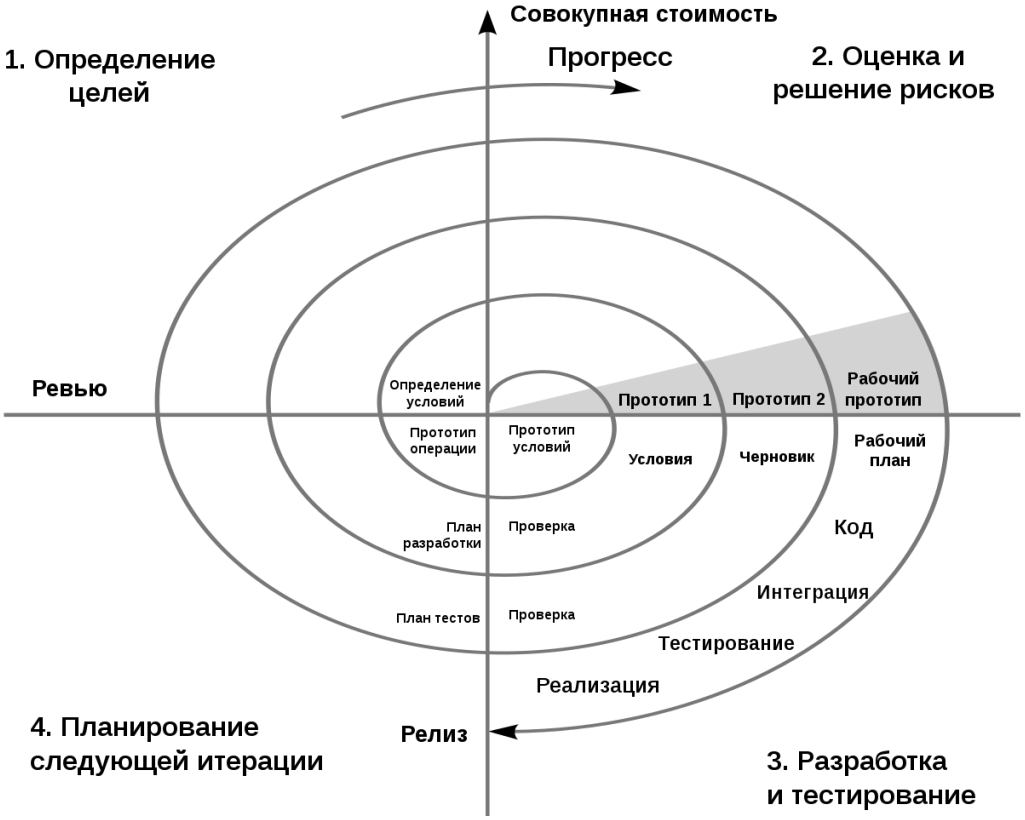

Рис. 6. Спиральная модель разработки программного продукта

Спиральная модель (рис. 6) является моделью разработки программного обеспечения, которая обеспечивает поддержку обработки рисков. В схематичном представлении данная методология – спираль с множеством петель. Количеств петель заранее неизвестно и варьируется от проекта к проекту. Каждая петля спирали называется фазой процесса разработки программного обеспечения. Радиус спирали в любой точке представляет расходы проекта до настоящего времени, а угловой размер представляет прогресс, достигнутый до настоящего времени в текущей фазе. Наиболее важной особенностью спиральной модели является обработка рисков после начала проекта. Такое разрешение рисков легче сделать, разработав прототип. Спиральная модель позволяет справляться с рисками, предоставляя возможности для создания прототипа на каждом этапе разработки программного продукта. К минусам спиральной модели можно отнести: большую зависимость от анализа рисков, сложность оценки временных интервалов проекта, дороговизну готового продукта [7].

Заключение

Таким образом, выбор методологии разработки является важной частью организации процесса разработки программного обеспечения. Грамотно выбранная методология позволяет достигнуть лучшего отношения затрат к полученному результату. Выбор зависит от специфики проекта, системы формирования бюджета, субъективных предпочтений и даже темперамента руководителя.

Исходя из разобранных методологий, можно заключить, что использование каскадной модели позволяет точно установить сроки и стоимость разработки, сама разработка проходит быстро, проектом управлять довольно просто. Каскадная модель дает отличные результаты только в проектах с четко и заранее определенными требованиями и способами их реализации. Использование данной модели не позволяет сделать шаг назад, процесс тестирования начинается только после полного завершения разработки продукта. Вносить изменения возможно только при полностью завершенном проекте. Каскадная модель используется в тех случаях, когда:

- все требования известны, понятны и зафиксированы. Противоречивых требований не имеется;
- нет проблем с доступностью программистов нужной квалификации;
- проект относительно небольшой.

В «гибкой» методологии разработки после каждой итерации заказчик может наблюдать результат и понимать, удовлетворяет он его или нет. Это одно из самых больших преимуществ гибкой модели. К недостаткам относится то, что из-за отсутствия конкретных формулировок результатов сложно оценить трудозатраты и стоимость, требуемые на разработку. Данная методология подходит для больших или нацеленных на длительный жизненный цикл проектов, постоянно адаптируемых к условиям рынка. Соответственно, в процессе реализации требования изменяются. Гибкая модель используется в тех случаях, когда:

- потребности пользователей постоянно меняются в динамическом бизнесе;
- существует необходимость внесения большого количества изменений за короткий срок и малый бюджет;
- существует необходимость обратной связи от пользователей по сильным и слабым сторонам продукта [8].

СПИСИОК ЛИТЕРАТУРЫ

1. Прогноз долгосрочного социально-экономического развития Российской Федерации на период до 2030 года. С. 72–73. [Электронный ресурс]. – URL: http://static.government.ru/media/files/41d457592e04b76338b7.pdf (дата обращения: 24.11.2020).

2. *Ехлаков Ю. П.* Планирование и организация вывода программного продукта на рынок: учебное пособие / Ю. П. Ехлаков. – Томск: Эль Контент, 2017. – 121 с.
3. Бессмертная классика Waterfall [Электронный ресурс]. – URL: https://worksection.com/blog/waterfall.html (дата обращения: 24.11.2020).
4. What is Rapid Application Development Model (RAD) [Электронный ресурс]. – URL: https://hackr.io/blog/rapid-application-development-model (дата обращения: 24.11.2020).
5. *Вольфсон Б. И.* Гибкое управление проектами и продуктами / Питер; Санкт-Петербург; 2014. – 144 с. ISBN 978-5-496-01323-9.
6. A Practical Guide to Seven Agile Methodologies, Part 2 [Электронный ресурс]. – URL: http://www.devx.com/architect/article/32836/1954 (дата обращения: 24.11.2020).
7. Andrew Stellman & Jennifer Greene Learning Agile: Understanding Scrum, XP, Lean, and Kanban // Publisher: O'Reilly Media; 1 edition – December 10, 2013. 420 pages.
8. *В.И. Грекул*, Н.Л. Коровкина, Ю.В. Куприянов Проектирование информационных систем. Практикум: Учебное пособие / В.И. Грекул, Н.Л. Коровкина, Ю.В. Куприянов – М.: Национальный Открытый Университет «ИНТУИТ» 2012. –187 с.

ИНФОРМАЦИЯ ОБ АВТОРАХ

Гидроец Михаил Олегович –
студент кафедры системного анализа и логистики
Санкт-Петербургский государственный университет аэрокосмического приборостроения
190000, Россия, Санкт-Петербург, ул. Большая Морская, д. 67, лит. А
E-mail: feston25@yandex.ru

Гришанова Лариса Иосифовна –
к.т.н., доцент кафедры системного анализа и логистики
Санкт-Петербургский государственный университет аэрокосмического приборостроения
190000, Россия, Санкт-Петербург, ул. Большая Морская, д. 67, лит. А

E-mail: grish@mail.ru

INFORMATION ABOUT THE AUTHORS

Gidroets Michael Olegovich –
student of the department of system analysis and logistics
Saint-Petersburg State University of Aerospace Instrumentation
SUAI, 67, Bolshaya Morskaia str., Saint-Petersburg, 190000, Russia
E-mail: feston25@yandex.ru

Grishanova Larisa Iosifovna –
PhD in Technical Sciences
Saint-Petersburg State University of Aerospace Instrumentation
SUAI, 67, Bolshaya Morskaia str., Saint-Petersburg, 190000, Russia
E-mail: grish@mail.ru