DOI: 10.17747/2618-947X-2024-3-270-279
УДК 338.984
JEL M15

Технологические особенности создания ИТ-продуктов в рамках выполнения ИТ-проектов

В.С. Николаенко[1, 2, 3]
[1] Томский государственный университет систем управления и радиоэлектроники (Томск, Россия)
[2] Томский политехнический университет (Томск, Россия)
[3] Сибирский государственный медицинский университет (Томск, Россия)

Аннотация
В статье рассматриваются свойства программ для ЭВМ, такие как инкрементность и высокая технологичность, а также технологические особенности управления ИТ-проектами. Целью статьи является анализ технологических особенностей создания программ для ЭВМ в ходе выполнения ИТ-проектов. На основании проведенного исследования было установлено, что для создания программ для ЭВМ применяют каскадные и гибкие концепции (Waterfall и Agile), порядка четырнадцати техник (XP, RUP, AUP, RAD, DSDM, Scrum, DAD, Kanban, Lean SD, FDD, MDD, DevOps, MSF и Oracle CDM) и четыре модели жизненного цикла (V-model, модель жизненного цикла Боема, итеративная и каскадная модели жизненного цикла). Кроме того, было обнаружено, что любой ИТ-проект, реализуемый по каскадной модели жизненного цикла, независимо от его масштаба, сложности, длительности, типа, способов управления и численности участников команды, проходит шесть фаз: начало ИТ-проекта, определение требований к создаваемой программе для ЭВМ, планирование, кодирование, тестирование и окончание ИТ-проекта. Полученные результаты позволили заключить, что концепции и техники создания программ для ЭВМ, а также модели жизненных циклов являются необходимыми знаниевыми компетенциями, которыми обязаны владеть все участники ИТ-проектов. Недостаточное владение такими компетенциями либо их отсутствие ставит под угрозу возможность достижения запланированных проектных целей, получения работоспособного программного кода, а также исполнения ковенант сделки.
Ключевые слова: ИТ-проект, управление проектами, Waterfall, Agile, Scrum.

Для цитирования:
Николаенко В.С. (2024). Технологические особенности создания ИТ-продуктов в рамках выполнения ИТ-проектов. *Стратегические решения и риск-менеджмент*, 15(3): 270–279. DOI: 10.17747/2618-947X-2024-3-270-279.

Благодарности
Работа выполнена в рамках государственного задания «Наука», проект FEWM-2023-0013.

Technological features of creating IT products within the framework of implementing IT projects

V.S. Nikolaenko[1, 2, 3]
[1] Tomsk State University of Control Systems and Radioelectronics (Tomsk, Russia)
[2] Tomsk Polytechnic University (Tomsk, Russia)
[3] Siberian State Medical University (Tomsk, Russia)

Abstract
The article examines the characteristics of ECM programs, such as incrementalism and high technology, as well as the technological features IT project management that create similar characteristics. The purpose of the article is to analyse the technological features of IT project management that arise during the creation of ECM programs. The study found that waterfall and flexible approaches, fourteen techniques and four lifecycle models are used to create ECMr programs. In addition, it was found that every IT project implemented according to the waterfall life cycle model, regardless of its size, complexity, duration, type, management methods and number of team members, goes through the six phases, such as starting an IT project, determining the requirements for the created ECM program, planning, coding, testing and closing the IT project. From the results obtained, it could be concluded that the concepts and techniques for creating ECM programs, as well as lifecycle models, are necessary knowledge competencies that all participants in IT projects must possess. Inadequate possession or lack of these skills jeopardises the ability to achieve the planned project objectives, to produce a working program code and also to fulfil the obligations of the transaction.
Keywords: IT project, Project Management, Waterfall, Agile, Scrum.

For citation:
Nikolaenko V.S. (2024). Technological features of creating IT products within the framework of implementing IT projects. *Strategic Decisions and Risk Management,* 15(3): 270-279. DOI: 10.17747/2618-947X-2024-3-270-279. (In Russ.)

Acknowledgment
The work was carried out within the framework of the state task 'Science', project FEWM-2023-0013.

在IT项目实施框架内创建IT产品的技术特点

V.S. Nikolaenko[1, 2, 3]
[1] 托姆斯克国立系统管理与无线电电子大学（俄罗斯，托木斯克）
[2] 托木斯克理工大学（俄罗斯，托木斯克）
[3] 西伯利亚国立医科大学（俄罗斯，托木斯克）

简介
本文讨论了计算机软件的特性，如增量和高可制造性，以及 IT 项目管理的技术特点。 本文旨在分析在 IT 项目过程中创建计算机程序的技术特点。研究发现，级联和敏捷概念（Waterfall和Agile）、约14种技术（XP、RUP、AUP、RAD、DSDM、Scrum、DAD、Kanban、Lean SD、FDD、MDD、DevOps、MSF 和 Oracle CDM）和4种生命周期模型（V 模型、Boehm 生命周期模型、迭代和级联生命周期模型）用于创建计算机软件。 此外，研究还发现，任何采用级联生命周期模式的信息技术项目，无论其规模、复杂程度、持续时间、类型、管理方法和团队成员数量如何，都要经历六个阶段：信息技术项目的开始、确定要创建的计算机程序的要求、规划、编码、测试和信息技术项目的结束。 研究结果使我们得出结论，计算机软件制作的概念和技术以及生命周期模型是信息技术项目的所有参与者都必须掌握的必要知识能力。如果不具备这些能力或缺乏这些能力，就有可能无法实现计划的项目目标，无法获得可行的软件代码，也无法履行交易契约。
关键词: IT 项目、项目管理、Waterfall, Agile, Scrum.

引用文本：
Nikolaenko V.S. (2024). 在IT项目实施框架内创建IT产品的技术特点。战略决策和风险管理, 15(3): 270–279. DOI: 10.17747/2618-947X-2024-3-270-279. (俄文)

致谢
这项工作是在国家任务 "科学 "项目 FEWM-2023-0013 框架内完成的。

Анализ трудов [О'Коннэл, 2005; Черников, Дашицыренов, 2017; Марченко и др., 2020; Базарова, Рочев, 2022] позволил установить, что основными технологическими свойствами программ для ЭВМ являются инкрементность и высокая технологичность.

Инкрементность программ для ЭВМ заключается в возможности добавления в программный код новых данных и команд с целью расширения функциональных возможностей и исправления программных ошибок (bug). Ярким примером инкрементности является компьютерная игра Cyberpunk 2077[1], выпуск которой состоялся в конце 2020 года. Несмотря на относительную давность релиза итоговой версии игры, организация-разработчик *CD Projekt RED* систематически выпускает обновления, улучшая технические характеристики, исправляя ошибки и добавляя новый контент. Например, в сентябре 2022 года разработчик выпустил патч 1.6, где был добавлен контент сериала Cyberpunk: Edgerunners, премьера которого состоялась осенью 2022 года на Netflix.

Инкрементность дает возможность декомпозировать желаемую программу для ЭВМ на отдельные пользовательские истории (user stories) и по частям поставлять их заинтересованным сторонам. Как отмечают в своих трудах К. Вигерс и Д. Битти, заинтересованные стороны очень часто предъявляют большое количество противоречивых пользовательских, функциональных и бизнес-требований, которые легко устраняются за счет инкрементности. В частности, она позволяет сначала создавать пользовательские истории, которые удовлетворяют требованиям всех сторон, а затем, от обновления к обновлению, постепенно устранять критические противоречия [Вигерс, Битти, 2022].

Стоит отметить, что свойство инкрементности радикально отличает программу для ЭВМ от работ, которые создаются в классических проектах (строительных, образовательных и др.). В частности, если отдельные части программы для ЭВМ могут разрабатываться параллельно, то в классических проектах желаемый результат получают только при выполнении определенной последовательности действий.

Еще одним свойством программ для ЭВМ является их высокая технологичность. Это означает, что к созданию программного кода могут привлекаться только те специалисты, которые обладают необходимыми профессиональными компетенциями. Например, программист[2] должен иметь минимальный уровень профессиональной квалификации, специалист по тестированию[3], администратор баз данных[4] и системный аналитик[5] – среднее специальное образование, специалист по дизайну графических пользовательских ин-

[1] https://www.cyberpunk.net/ru/ru/.
[2] Профессиональный стандарт 06.001 «Программист». https://clck.ru/PaFBJ.
[3] Профессиональный стандарт 06.004 «Специалист по тестированию в области информационных технологий». https://clck.ru/PaFa5.
[4] Профессиональный стандарт 06.011 «Администратор баз данных». https://clck.ru/qNbpz.
[5] Профессиональный стандарт 06.022 «Системный аналитик». https://clck.ru/PaFVa.

Таблица 1
Основные достоинства и недостатки концепции каскадного создания программ для ЭВМ
Table 1
The main advantages and disadvantages of the cascade approach to programming for ECM

Достоинства/недостатки	Комментарий
Основные достоинства концепции каскадного создания программ для ЭВМ	
Дифференциация фаз создания программ для ЭВМ	Ясные границы фаз дают возможность определить их точные даты начала и окончания, что не только повышает лояльность заказчиков, но и является одним из существенных условий при заключении контрактов (ст. 432 ГК РФ[a])
Твердая цена (Fixed Price)	Определение твердой цены повышает лояльность заказчиков программ для ЭВМ и конкурентоспособность организаций, занятых их созданием
Качество проектной документации	Как правило, в процессе создания программ для ЭВМ принимают участие различные работники, поэтому качественная проектная документация необходима для их слаженной и синхронной работы
Гармоничная интеграция новых участников в команду ИТ-проекта	Каждая фаза завершается выпуском комплекта проектной документации, достаточной для продолжения разработки другой командой
Основные недостатки концепции каскадного создания программ для ЭВМ	
Сложность изменения ранее утвержденных пользовательских, функциональных и бизнес-требований	Как правило, изменение требований возможно только после создания определенного ИТ-результата
Продолжительность создания программы для ЭВМ	Каскадная разработка программ для ЭВМ является длительным процессом, что повышает вероятность наступления таких рисков, как изменение норм действующего законодательства, трансформация бизнес-структуры и интересов заказчика, уход ключевых работников и др.
Отклонение от запланированных сроков	Создание программ для ЭВМ редко укладывается в запланированные сроки. В частности, в аналитических докладах The CHAOS Manifesto сказано, что среднее отклонение от запланированных сроков в ИТ-проектах составляет 89%[b]

[a] Гражданский кодекс Российской Федерации от 30.11.1994 № 51-ФЗ. https://clck.ru/MsKTf.
[b] The Standish Group International. https://clck.ru/3Div55.

терфейсов[6] – профессиональную подготовку сроком до одного года, а руководитель проекта в области ИТ[7] – высшее образование по программе бакалавриата.

В работе [Черников, Дашицыренов, 2017] подчеркивается, что успех ИТ-проектов во многом зависит от профессиональных компетенций и опыта программистов. Несоответствие уровню квалификации неминуемо будет приводить к многочисленным дефектам и снижать работоспособность разрабатываемых программ.

Высокая технологичность программ для ЭВМ также проявляется в возможности создания программного кода дистанционно [Конобевцев и др., 2019]. Дистанционная форма организации команды ИТ-проекта имеет ряд преимуществ перед классическими проектами. Например, «дистант» позволяет привлекать квалифицированных работников из разных регионов и стран, производить оплату по факту выполненных работ, а также уменьшать себестоимость разработки программного кода, исключив расходы, связанные с арендой офисных помещений, оплатой электроэнергии, интернета, коммунальных услуг и др.

В связи с этим целью настоящей статьи является анализ технологических особенностей создания программ для ЭВМ в ходе выполнения ИТ-проектов. Для достижения поставленной цели автором настоящей статьи были решены следующие задачи:

- выявлены достоинства и недостатки концепций создания программ для ЭВМ;
- проанализированы техники создания программ для ЭВМ (XP, RUP, AUP, RAD, DSDM, Scrum и др.);
- идентифицированы основные модели жизненных циклов (далее – ЖЦ) ИТ-проектов.

Инкрементность и высокая технологичность программ для ЭВМ простимулировали развитие различных концепций создания программ для ЭВМ, таких как Waterfall и Agile.

Концепция каскадного создания программ для ЭВМ (Waterfall). Считается, что концепция каскадного (классического, водопадного) создания программ для ЭВМ была разработана в 1970 году американским ученым У.У. Ройсом [Royce, 1970]. По мнению Ройса, процесс создания программного кода похож на непрерывный поток воды, где каждая фаза продолжает предыдущую и не начинается до тех пор, пока прошлая не заканчивается. Основные достоинства и недостатки концепции каскадного создания программ для ЭВМ представлены в табл. 1.

Концепция гибкого создания программ для ЭВМ (Agile). Гибкая концепция создания программ для ЭВМ была создана в феврале 2001 года К. Беком, М. Бидлом, Э. В. Беннекумом и др. [Макконнелл, 2021]. Используя свойство инкрементности, авторы сформулировали основные принципы гибкого управления ИТ-проектами, отличные от каскадной концепции. Примеры принципов концепции гибкого создания программ для ЭВМ представлены в табл. 2.

Стоит отметить, что в отечественной литературе концепцию Agile принято называть методологией гибкой раз-

[6] Профессиональный стандарт 06.025 «Специалист по дизайну графических пользовательских интерфейсов». https://clck.ru/PaFGs.
[7] Профессиональный стандарт 06.016 «Руководитель проектов в области информационных технологий». https://clck.ru/PaFDk.

Таблица 2
Принципы концепции гибкого создания программ для ЭВМ
Table 2
Principles of the flexible programming approach to ECM

Название принципа работы	Описание принципа работы
Частые и непрерывные поставки частей программы для ЭВМ	Частые и непрерывные поставки программы для ЭВМ важны для заказчиков и пользователей, т. к. благодаря им заинтересованные стороны могут уточнить пользовательские, функциональные и бизнес-требования, а также сократить срок возврата инвестиций
Максимальная открытость для возможности изменения требований	Наивысшим приоритетом для концепции гибкого создания программ для ЭВМ является удовлетворенность заинтересованных сторон; по этой причине Agile стремится по возможности учесть все пользовательские, функциональные и бизнес-требования
Гибкость процессов	Принцип максимальной открытости для возможности изменения требований на любой фазе ЖЦ ИТ-проекта подразумевает, что процессы обязаны оперативно адаптироваться под новые и/или измененные требования
Систематическая поставка действующего ИТ-результата	Для нивелирования таких рисков, как ошибки, дефекты, неточности и уязвимости программного кода, несоответствие ожиданиям заинтересованных сторон, судебные споры и др., Agile предусматривает систематическую поставку действующего ИТ-результата
Максимальная вовлеченность заинтересованных сторон	Сотрудничество и открытость коммуникаций в Agile важнее иерархии и контрактных ограничений, потому процесс создания ценной программы для ЭВМ для всех заинтересованных сторон возможен, если были учтены все интересы и мнения
Самоорганизация команды ИТ-проекта	Agile не формализует конкретные процессы реализации ИТ-проекта, а только задает общий вектор разработки: планирование, анализ требований, проектирование, программирование, тестирование, документирование. Например, декомпозиция пользовательских историй осуществляется силами команды ИТ-проекта. Под пользовательскими историями понимается описание требований к разрабатываемой программе для ЭВМ. В качестве примера пользовательской истории можно привести кейс, когда клиент банка хочет получать сообщения об изменении статуса заявки на кредит, чтобы оперативно распоряжаться денежными средствами. Также стоит отметить, что во время спринта[a] участники ИТ-проекта самостоятельно принимают решения, какие запланированные работы и в какой последовательности будут выполняться
Общение лицом к лицу	В Agile считается, что самый эффективный и результативный способ взаимодействия между заинтересованными сторонами – это личное общение
Результативность	Главной мерой прогресса являются работоспособные части программы для ЭВМ, которые систематически поставляются заинтересованным сторонам
Постоянное профессиональное развитие участников ИТ-проекта	Максимальная открытость для возможности изменения пользовательских, функциональных и бизнес-требований стимулирует участников команды ИТ-проекта к непрерывному профессиональному развитию и изучению новых особенностей создания программ для ЭВМ
Keep it short and simple (KISS)	Э. Рэймонд в своих трудах отмечает, что при проектировании программ для ЭВМ необходимо обеспечивать максимальную простоту и прозрачность программного кода [Raymond, 2003]

[a] Спринт – временной интервал, в течение которого команда ИТ-проекта выполняет запланированный объем работы.

работки программного обеспечения [Обри, 2019]. Однако, согласно классическому толкованию, под методологией понимается совокупность методов, средств и технологий познания, применяемых с целью организации и построения исследования [Лузгина, 2018; Смагина, 2020]. По мнению автора настоящей статьи, использование понятия «концепция» является более приемлемым, так как Agile представляет собой совокупность принципов, подходов, лучших практик, идей и способов создания программ для ЭВМ. В связи с этим под Agile необходимо понимать концепцию гибкого создания программ для ЭВМ, включающую в себя совокупность специальных принципов, подходов, лучших практик, идей и способов достижения проектных целей.

Помимо технологических свойств, которые стали причинами дифференциации концепций создания программ для ЭВМ, на ход реализации ИТ-проектов также оказывают влияние неопределенность и изменчивость пользовательских, функциональных и бизнес-требований, интеллектуальный труд, различные стили управления, ограниченные ресурсы, кросс-коммуникация с заинтересованными сторонами и др. Учет этих факторов в управлении простимулировал развитие множества различных техник создания программ для ЭВМ: XP, RUP, AUP, RAD, DSDM, Scrum и др.

Перечень техник создания программ для ЭВМ представлен в табл. 3.

Технологические свойства, дифференциация концепций и множество техник создания программ для ЭВМ оказали влияние на логические связи между фазами в ИТ-проектах, что простимулировало создание различных моделей жизненных циклов.

Жизненный цикл проекта – это совокупность фаз, через которые проходит проект от старта до завершения. Под фазой проекта (стадией проекта) понимается совокупность операций, которые завершаются достижением запланированных результатов. Фаза проекта, как правило, заканчивается контрольной точкой (вехой) [Исаев и др., 2021]. Фазы проекта могут быть последовательными, итеративными и/или накладываться друг на друга [Полонский, Васильев, 2018]. Примеры моделей ЖЦ, актуальных для ИТ-проектов, представлены в табл. 4.

Таблица 3
Техники создания программ для ЭВМ
Table 3
Techniques for writing programs for ECM

Название техники	Описание техники
Экстремальное программирование (eXtreme Programming, XP)	Название основывается на идее использовать только лучшие практики создания программного кода, выводя процесс его создания на новый уровень – экстремальный. Например, при проверке созданного программного кода рекомендуется привлекать одновременно двух программистов, для того чтобы один был занят созданием программы, а его напарник – ее проверкой [Бек, 2003]. Эту лучшую практику принято называть парным программированием. Среди достоинств техники XP следует отметить оперативное получение первой версии программы для ЭВМ, что дает возможность пользователям приступить к ее апробации
IBM Rational Unified Process (RUP)	Включает в себя 9 процессов и 4 фазы создания программы для ЭВМ. К основным процессам относятся: бизнес-моделирование, управление требованиями, анализ и проектирование, реализация, тестирование, развертывание, управление проектными работами, управление изменениями и инфраструктурой (внутренняя среда создания программного кода). Среди фаз выделяются: начальная фаза, уточнение, конструирование и внедрение
Agile Unified Process (AUP)	AUP – это упрощенная версия RUP. AUP, включающая в себя 7 процессов: моделирование, реализация, тестирование, развертывание, управление конфигурациями, управление проектом и создание инфраструктуры [Edeki, 2013]
Rapid Application Development (RAD)	RAD рекомендуется применять для краткосрочных ИТ-проектов, для которых характерны сжатые сроки, ограниченный бюджет, небольшой объем работ, наличие графического интерфейса и низкая вычислительная сложность [Beynon-Davies et al., 1999]
Dynamic Systems Development Method (DSDM)	DSDM базируется на концепции быстрой разработки приложений RAD. Инструментарий базируется на своем собственном ЖЦ, который состоит следующих фаз: оценка возможности технической реализации; экономическое обоснование; создание функциональной модели; проектирование; разработка.
Scrum	Scrum состоит из таких элементов, как роли, артефакты и процессы. В Scrum принято выделять три основных роли – это менеджер продукта (product owner), scrum-мастер и команда проекта. Артефактами принято называть приоритизированный перечень требований (product backlog), перечень требований, которые были отобраны на спринт (sprint backlog) и инкремент программного кода. Процессы в Scrum связаны с коммуникациями. В частности, короткое собрание команды проекта с целью синхронизации работ (scrum meeting), спринт-планирования (собрание команды проекта с целью декомпозиции user story) и проведения ретроспективного анализа [Кон, 2011]
Disciplined Agile Delivery (DAD)	Инструментарий DAD, разработанный IBM, основан на Scrum. Основным отличием является расширенный ЖЦ ИТ-проекта, который начинается классической инициацией проекта и заканчивается использованием полученного ИТ-результата пользователем
Kanban	Kanban основан на японской технологии бережливого производства – «точно в срок». Главным достоинством инструментария является равномерная нагрузка между участниками команды проекта. Работы (tasks) по мере их поступления заносятся в отдельный список (pull)
Бережливая разработка программного обеспечения (Lean SD)	Lean SD был разработан в [Поппендик, Поппендик, 2010]. Техника основана на традиционных принципах бережливого производства, таких как исключение потерь и пауз в процесс разработки, акцент на обучение, предельно отсроченное принятие решений, основанное на фактах, предельно быстрая доставка работающего ИТ-результата заказчику, мотивация команды проекта, интегрирование, целостное видение
Feature Driven Development (FDD)	FDD для удобства контроля за ходом создания ИТ-результата использует идеальную модель трудозатрат, где на анализ предметной области отводится 1%, дизайн – 40%, проверку и доработку дизайна 3%, создание программного кода – 45%, тестирование и доработку программного кода – 10%, внедрение – 1%
Model Driven Development (MDD)	Для этой техники создания программ для ЭВМ характерно абстрактное описание желаемого результата, где могут быть не учтены некоторые аспекты. Это делается для того, чтобы упростить процесс проектирования и документирования
Development and Operations (DevOps)	Главной особенностью DevOps является максимальная интеграция между программистами и специалистами по информационно-технологическому обслуживанию информационных систем
Microsoft Solutions Framework (MSF)	Корпорация Microsoft предлагает MSF для управления работой команды программистов уровня организации. По мнению авторов MSF, существует множество причин, приводящих к неудачам: ошибки в прогнозах, изменение требований, неточные спецификации и др. MSF состоит из 3 моделей: модель команды, модель проектной группы и модель процесса проектирования
Oracle Custom Development Method (Oracle CDM)	Oracle CDM состоит из 3 моделей: классическая модель (Classic), быстрой разработки (Fast Track) и облегченной разработки (Lite). Для каждой модели существует свой набор фаз и процессов. Например, Classic имеет 6 фаз и 11 процессов. Classic применяется для долгосрочных (длительность более 6 месяцев) и сложных ИТ-проектов. Fast Track ориентирован на заказные разработки, длительность которых не превышает 6 месяцев. ЖЦ быстрой разработки состоит из 3 фаз (моделирование требований, проектирование и создание, внедрение) и 11 процессов. Lite применяется для краткосрочных проектов и состоит из 2 фаз (прототипирование и построение, внедрение) и 9 процессов

Таблица 4
Модели жизненных циклов ИТ-проектов
Table 4
Lifecycle models for IT projects

Модель ЖЦ	Описание модели ЖЦ
V-образная	Фазы ИТ-проекта следуют друг за другом последовательно в строго определенном порядке [Balaji, Sundararajan, 2012]. Тестирование требований в ЖЦ происходит слишком поздно, вследствие чего невозможно внести изменения, не повлияв на план проектных работ
Спиральная	Спиральная модель делает упор на анализ, проектирование и управление рисками [Boehm, 1981]. На каждом витке спирали выполняется создание новой версии программы для ЭВМ. Один виток спирали представляет собой законченный проектный цикл, основывающийся на каскадной концепции
Итеративная	Принцип работы итеративной модели ЖК аналогичен спиральной. Отличие заключается в том, что в итеративной модели результатом работ является макет-прототип фрагмента будущей программы для ЭВМ. В процессе разработки итерации выполняются, пока макет-прототип не приобретет все необходимые свойства в соответствии с ТЗ. Как правило, итеративная модель применима для краткосрочных (менее 2 месяцев) и среднесрочных (2–6 месяцев) ИТ-проектов
Каскадная	Фазы ЖЦ идут друг за другом последовательно. Несмотря на ряд недостатков, каскадная модель ЖЦ может быть применима для любого типа, размера и сложности ИТ-проекта. Универсальными считаются следующие фазы: начало проекта, организация и подготовка, выполнение работ и окончание проекта

Для правильного выбора модели ЖЦ ИТ-проекта обычно руководствуются четырьмя критериями: стоимостью, риском, качеством и скоростью разработки. Эти критерии взаимосвязаны, поэтому нельзя достичь всех четырех целей одновременно. Например, если дату получения итоговой версии программы необходимо форсировать, то это потребует привлечения дополнительных кадровых ресурсов, а значит, приведет к увеличению стоимости ИТ-проекта.

Анализ национальных стандартов позволил установить, что отечественная проектная практика создания программ для ЭВМ базируется на каскадной модели ЖЦ[8], в которой выделяется 8 фаз[9]. Примеры фаз отечественной каскадной модели ЖЦ создания программ для ЭВМ представлены в табл. 5.

На основании проведенного анализа национальных стандартов, систематизирующих способы создания программ

Таблица 5
Фазы каскадной модели ЖЦ создания программ для ЭВМ согласно отечественным стандартам
Table 5
Phases of the cascade model lifecycle for creating ECM programs according to national standards

Модель ЖЦ	Описание
Формирование требований (предпроектная фаза)	Претендент в подрядчики (исполнители) проводит обследование существующей инфраструктуры заказчика, идентифицирует текущие проблемы, угрозы и возможности, обосновывает необходимость создания ИТ-результата, выявляет пользовательские, функциональные и бизнес-требования
Разработка концепции автоматизированной системы	Претендент в подрядчики (исполнители) создает модель As Is («Как есть»), например в виде IDEF0-модели, прорабатывает варианты возможных ИТ-решений, которые удовлетворяют пользовательским, функциональным и бизнес-требованиям, и разрабатывает модель «To Be» («Как должно быть») [Коваленко, Чокла, 2020]
Разработка ТЗ	В силу ГОСТ 19.101-77[a] и ГОСТ 15.016–2016 [b] под техническим заданием понимается документ, где формализованы назначение и область применения программы для ЭВМ, технические и специальные требования. В литературе также можно встретить другие название ТЗ. Например, в [Вигерс, Битти, 2022] техническое задание называется спецификацией требований к программе для ЭВМ
Разработка эскизного проекта	Согласно ГОСТ 2.119-2013[c] и ГОСТ 2.103-2013[d] под эскизным проектом понимается совокупность проектных документов, которые содержат принципиальные решения, дающие общее и предварительное представление о назначении, устройстве, принципе работы программы для ЭВМ, а также данные, определяющие их основные параметры. Эскизный проект позволяет выбрать подходящее ИТ-решение для последующей разработки
Разработка технического проекта	В соответствии с ГОСТ 2.120-2013[f] технический проект – это документ, где в соответствии с ТЗ и эскизом проекта закрепляются окончательные технические решения. Стоит отметить, что концепция, ТЗ, эскизный и технический проект являются частями проектной документации. Под проектной документацией понимается документация в текстовой и графической форме, содержащая сведения, необходимые для разработки, сопровождения и эксплуатации ИТ-результата
Разработка рабочей документации	Рабочая документация – это материалы в текстовой и графической форме, в соответствии с которыми осуществляется создание ИТ-результата

[8] Национальный стандарт Российской Федерации. Информационные технологии. Системная и программная инженерия. Управление жизненным циклом. Ч. 2. Руководство по применению ИСО/МЭК 15288. ГОСТ Р 57102-2016/ISO/IEC TR 24748-2:2011. М., Стандартинформ, 2016.

[9] Государственный стандарт Союза ССР. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания. ГОСТ 34.601-90. М., Стандартинформ, 1992.

Таблица 5 – *окончание*
Table 5 – *remainder*

Модель ЖЦ	Описание
Ввод в действие	На этой фазе осуществляются монтажные и пусконаладочные работы, проводятся предварительные испытания и опытная эксплуатация, подготавливаются и обучаются работники, осуществляется приемка созданной программы для ЭВМ. В силу п. 1.4 ГОСТ 34.201-89[g] разрабатываются акты завершения работ, приемки в опытную эксплуатацию, приемки в промышленную эксплуатацию и др.
Сопровождение	Для этой фазы характерно выполнение работ (оказание услуг, поставки товаров) в соответствии с гарантийными обязательствами, а также послегарантийное обслуживание

[a] Государственный стандарт Союза ССР. Единая система программной документации. Виды программ и программных документов. ГОСТ 19.101–77. М.. Стандартинформ, 1980.
[b] Межгосударственный стандарт. Система разработки и постановки продукции на производство. Техническое задание. Требования к содержанию и оформлению. ГОСТ 15.016–2016. М., Стандартинформ, 2020.
[c] Межгосударственный стандарт. Единая система конструкторской документации. Эскизный проект. ГОСТ 2.119–2013. М., Стандартинформ, 2018.
[d] Межгосударственный стандарт. Единая система конструкторской документации. Стадии разработки. ГОСТ 2.103–2013. М., Стандартинформ, 2019.
[e] Государственный стандарт Союза ССР. Обеспечение технологичности конструкции изделий. Общие требования. ГОСТ 14.201–83. М., Стандартинформ, 2009.
[f] Межгосударственный стандарт. Единая система конструкторской документации. Технический проект. ГОСТ 2.120–2013. М., Стандартинформ, 2015.
[g] Государственный стандарт Союза ССР. Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплектность и обозначение документов при создании автоматизированных систем. ГОСТ 34.201–89. М., Стандартинформ, 1989.

для ЭВМ, автор настоящей статьи выделяет шесть фаз ЖЦ, актуальных для любого ИТ-проекта:

- начало ИТ-проекта,
- определение требований к создаваемой программе для ЭВМ,
- планирование,
- кодирование,
- тестирование,
- окончание ИТ-проекта (табл. 6).

На основании анализа технологических особенностей управления ИТ-проектами можно заключить следующее.

- Основными особенностями создания программ для ЭВМ в ходе выполнения ИТ-проектов являются инкрементность и высокая технологичность [Николаенко, 2020; Nikolaenko, Sidorov, 2023].
- Для создания программ для ЭВМ применяют каскадные и гибкие концепции (Waterfall и Agile), порядка четырнадцати техник (XP, RUP, AUP, RAD, DSDM,

Таблица 6
Фазы жизненного цикла ИТ-проекта
Table 6
Phases of the IT project lifecycle

Фаза ЖЦ	Описание
Начало ИТ-проекта	Для этой фазы характерно проведение переговоров между заинтересованными сторонами, обычно между заказчиком и подрядчиком (исполнителем, поставщиком), в рамках которых утверждаются требования к программе для ЭВМ, даты начала и окончания работ, цена, способы нивелирования и ослабления коммерческих, комплаенс- и проектных рисков и др. Как правило, по окончании фазы между сторонами заключается гражданско-правовой договор (контракт)
Определение требований к создаваемой программе для ЭВМ	Выявленные пользовательские, функциональные и бизнес-требования формализуются в ТЗ
Планирование	На основании ТЗ и коммуникаций с заинтересованными сторонами руководитель ИТ-проекта определяет концепцию и технику создания программы для ЭВМ, подготавливает необходимые ресурсы и основные проектные документы
Кодирование	Фаза предусматривает создание программного кода в соответствии с пользовательскими, функциональными и бизнес-требованиями
Тестирование	На этой фазе проводятся испытания созданного программного кода для установления соответствия между реальным поведением разработанных алгоритмов с их ожидаемым поведением[a]. Согласно п. 4.2 ГОСТ 34.603-92[b] для проведения испытаний необходима такая документация, как ТЗ, акт приемки в опытную эксплуатацию, рабочие журналы опытной эксплуатации и др.
Окончание ИТ-проекта	С учетом комплаенс-особенностей на этой фазе осуществляется приемка созданной программы для ЭВМ с подписанием соответствующих актов

[a] Программная инженерия. Руководство к своду знаний по программной инженерии (Software Engineering – Guide to the Software Engineering Body of Knowledge (SWEBOK)). ISO/IEC TR 19759:2005. 2005.
[b] Государственный стандарт Союза ССР. Информационная технология. Виды испытаний автоматизированных систем. ГОСТ 34.603–92. М., Стандартинформ, 1993.

Scrum, DAD, Kanban, Lean SD, FDD, MDD, DevOps, MSF и Oracle CDM) и более четырех моделей ЖЦ (V-model, модель ЖЦ Боема, итеративная и каскадная модели ЖЦ).

- Любой ИТ-проект, реализуемый согласно каскадной модели ЖЦ, независимо от его масштаба, сложности, длительности, типа и способов управления, проходит шесть фаз: начало ИТ-проекта, определение требований к создаваемой программе для ЭВМ, планирование, кодирование, тестирование и окончание ИТ-проекта.

На основании сказанного также можно заключить, что концепции и техники создания программ для ЭВМ, а также модели ЖЦ являются необходимыми знаниевыми компетенциями, которыми обязаны владеть все участники ИТ-проектов. Недостаточное владение либо отсутствие этих компетенций ставит под угрозу возможность достижения запланированных проектных целей, получения работоспособного программного кода, а также исполнения ковенант сделки, что, в свою очередь, может спровоцировать последующую материализацию комплаенс- и проектных рисков.

Литература

Базарова А.М., Рочев К.В. (2022). IT-технологии в управлении производственными процессами на предприятиях топливно-энергетического комплекса. В: *Управление устойчивым развитием топливно-энергетического комплекса-2021:* материалы II Всероссийской научно-практической конференции. Ухта, 97–103.

Бек К. (2003). *Экстремальное программирование: разработка через тестирование*. СПб., Питер.

Вигерс К., Битти Д. (2022). *Разработка требований к программному обеспечению.* СПб., БХВ.

Исаев Е.А., Первухин Д.В., Рытиков Г.О., Филюгина Е.К., Айрапетян Д.А. (2021). Оценка эффективности информационных систем. *Бизнес-информатика,* 15(1): 19–29.

Коваленко В.В., Чокла Д.С. (2020). Трансформация процессов и организационной структуры WEB-системы модельного бизнеса по результатам бизнес-анализа. В: *Современные технологии в науке и образовании-2020:* III международный научно-технический форум, сборник трудов. Рязань, Book Jet, 195–198.

Кон М. (2011). *Scrum. Гибкая разработка ПО.* М., Вильямс.

Конобевцев Ф.Д., Ласс Н.А., Гурова Е.В., Романова И.А. (2019). Удаленная работа: технологии и опыт организации. *Вестник университета,* 7: 9–16.

Лузгина Я.А. (2018). Концепция, методология и методы информационного обеспечения инновационной деятельности. В: *Проблемы развития экономики и предпринимательства,* материалы 16-й Всероссийской научно-практической конференции: 58–64.

Макконнелл С. (2021). *Еще более эффективный Agile.* СПб., Питер.

Марченко Д.С., Григорьевых А.В., Рочев К.В. (2020). Информационная система хранения авторотационных данных. *Информационные технологии в управлении и экономике,* 3(20): 21–39.

Николаенко В.С. (2020). Риск, риск-менеджмент и неопределенность: уточнение понятий. *Государственное управление. Электронный вестник,* 81: 91–119.

О'Коннэл Ф. (2005). *Как успешно руководить проектами. Серебряная пуля.* М., Кудиц-Образ.

Обри К. (2019). *Все об Agile. Искусство создания эффективной команды.* М., Эксмо.

Полонский А.Ю., Васильев М.М. (2018). Анализ методологий разработки программного обеспечения. *Аллея науки,* 6(22): 1084–1094.

Поппендик М., Поппендик Т. (2010). *Бережливое производство программного обеспечения: от идеи до прибыли.* М., Вильямс.

Смагина С.М. (2020). О понятиях «метод научного исследования», «методология научного исследования» и «логика научного исследования». В: *Логика и методология научных исследований:* сборник научных статей и докладов международной научно-практической конференции. Орел, Среднерусский институт управления – филиал РАНХиГС, 112–116.

Черников Б.В., Дашицыренов З.Д. (2017). Анализ современных методов управления качеством и их применение к области высшего образования. В: *Управление развитием крупномасштабных систем MLSD'2017:* материалы 10-й международной конференции. М., Институт проблем управления им. В. А. Трапезникова, РАН, 217–219.

Balaji S., Sundararajan M.M. (2012). Waterfall Vs V-model Vs Agile: A comparative on SDLC. *International Journal of Information Technology and Business Management,* 2(1): 26−30.

Beynon-Davies P., Carne C., Mackay H., Tudhope D. (1999). Rapid application development (RAD): An empirical review. *European Journal of Information Systems,* 8: 211–223.

Boehm B.W. (1981). *Software engineering economics.* Englewood Cliffs, NJ, Prentice-Hall.

Edeki C. (2013). Agile unified process. *International Journal of Computer Science and Mobile Applications,* 1(3): 13–17.

Nikolaenko V., Sidorov A. (2023). Analysis of 105 IT project risks. *Journal of Risk and Financial Management,* 16(1), 33: 1–20.

Raymond E. (2003). *The art of Unix programming.* Addison-Wesley.

Royce W.W. (1970). *Managing the development of large software systems.*

References

Bazarova A.M., Rochev K. V. (2022). IT-technologies in the management of production processes at enterprises of the fuel and energy complex. In: *Management of sustainable development of the fuel and energy complex-2021.* Materials of the II All-Russian Scientific and Practical Conference. Ukhta, 97-103. (In Russ.)

Beck K. (2003). *Extreme programming: Development through testing.* St. Petersburg, Piter. (In Russ.)

Vigers K., Beatty D. (2022). *Development of software requirements.* St. Petersburg, BHV. (In Russ.)

Isaev E.A., Pervukhin D.V., Rytikov G.O., Filyugina E.K., Hayrapetyan D.A. (2021). Evaluation of the effectiveness of information systems. *Business Informatics,* 15(1): 19-29. (In Russ.)

Kovalenko V.V., Chokla D.S. (2020). Transformation of processes and organizational structure of a WEB-based model business system based on the results of business analysis. In: *Modern technologies in science and education-2020,* III International Scientific and Technical Forum, collection of works. Ryazan, Book Jet, 195-198. (In Russ.)

Con M. (2011). *Scrum. Flexible software development.* Moscow, Williams. (In Russ.)

Konobevtsev F.D., Lass N.A., Gurova E.V., Romanova I.A. (2019) Remote work: Technologies and experience of organization. *Bulletin of the University,* 7: 9-16. (In Russ.)

Luzgina Ya.A. (2018) Concept, methodology and methods of information support of innovation activity. In: *Materials of the 16th All-Russian Scientific and Practical Conference «Problems of economic development and entrepreneurship»,* 58-64. (In Russ.)

McConnell S. (2021). *Even more efficient Agile.* St. Petersburg, Piter. (In Russ.)

Marchenko D.S., Grigoriev A.V., Rochev K.V. (2020). Information system for storing autorotation data. *Information Technologies in Management and Economics,* 3(20): 21-39. (In Russ.)

Nikolaenko V. (2020). Risk, risk management and uncertainty: Clarifying the concepts. *Public Administration. Electronic Bulletin,* 81: 91-119. (In Russ.)

O'Connell F. (2005). *How to successfully manage projects. Silver bullet.* Moscow, Kudits-Obraz. (In Russ.)

Aubrey K. (2019). *All about Agile. The art of creating an effective team.* Moscow, Eksmo. (In Russ.)

Polonsky A.Yu., Vasiliev M.M. (2018). Analysis of software development methodologies. *Alley of Science,* 6(22): 1084-1094. (In Russ.)

Poppendik M., Poppendik T. (2010). *Lean software production: From idea to profit.* Moscow, Williams. (In Russ.)

Smagina S.M. (2020). On the concepts of 'method of scientific research', 'methodology of scientific research' and 'logic of scientific research'. In: *Logic and methodology of scientific research:* Collection of scientific articles and reports of the International Scientific and Practical Conference. Orel, The Central Russian Institute of Management, RANEPA, 112-116. (In Russ.)

Chernikov B.V., Dashitsyrenov Z.D. (2017). Analysis of modern methods of quality management and their application to the field of higher education. In: *Managing the development of large-scale systems MLSD'2017:* Proceedings of the 10th International Conference. Moscow, V.A. Trapeznikov Institute of Management Problems, RAS, 217-219. (In Russ.)

Balaji S., Sundararajan M.M. (2012). Waterfall Vs V-model Vs Agile: A comparative on SDLC. *International Journal of Information Technology and Business Management,* 2(1): 26-30.

Beynon-Davies P., Carne C., Mackay H., Tudhope D. (1999). Rapid application development (RAD): An empirical review. *European Journal of Information Systems,* 8: 211-223.

Boehm B.W. (1981). *Software engineering economics.* Englewood Cliffs, NJ, Prentice-Hall.

Edeki C. (2013). Agile unified process. *International Journal of Computer Science and Mobile Applications,* 1(3): 13-17.

Nikolaenko V., Sidorov A. (2023). Analysis of 105 IT project risks. *Journal of Risk and Financial Management,* 16(1), 33: 1-20.

Raymond E. (2003). *The art of Unix programming.* Addison-Wesley.

Royce W.W. (1970). *Managing the development of large software systems*.

Информация об авторе

Валентин Сергеевич Николаенко

Кандидат экономических наук, доцент кафедры автоматизации обработки информации Томского государственного университета систем управления и радиоэлектроники (Томск, Россия); доцент Бизнес-школы Томского политехнического университета (Томск, Россия); доцент кафедры экономики, социологии, политологии и права Сибирского государственного медицинско-

го университета (Томск, Россия). ORCID: 0000-0002-1990-4443; Web of Science Researcher ID: J-8521-2015; SPIN: 9301-1835; Author ID: 745788; IRID: 283767926; Scopus Author ID: 57193434445.
Область научных интересов: риск-менеджмент, национальная безопасность, экономическая безопасность, информационное право и защита интеллектуальной собственности, гражданское право, управление проектами.
valentin.s.nikolaenko@tusur.ru

About the author

Valentin S. Nikolaenko
Candidate of economic sciences, associate professor at the Department of Automation of Information Processing, Tomsk State University of Control Systems and Radioelectronics (Tomsk, Russia); associate professor at the Business School of Tomsk Polytechnic University (Tomsk, Russia); associate professor at the Department of Economics, Sociology, Political Science and Law of Siberian State Medical University (Tomsk, Russia). ORCID: 0000-0002-1990-4443; Web of Science Researcher ID: J-8521-2015; SPIN: 9301-1835; Author ID: 745788; IRID: 283767926; Scopus Author ID: 57193434445.
Research interests: risk-management, national security, economic security, information law and intellectual property protection, civil law, project management.
valentin.s.nikolaenko@tusur.ru

作者信息

Valentin S. Nikolaenko
经济学副博士, 托姆斯克国立系统管理与无线电电子大学（俄罗斯，托木斯克）信息处理自动化系教授; 托木斯克理工大学（俄罗斯，托木斯克）商学院副教授; 西伯利亚国立医科大学（俄罗斯，托木斯克）经济学、社会学、政治学和法学系副教授。ORCID: 0000-0002-1990-4443; Web of Science Researcher ID: J-8521-2015; SPIN: 9301-1835; Author ID: 745788; IRID: 283767926; Scopus Author ID: 57193434445.
科学兴趣领域: 风险管理、国家安全、经济安全、信息法和知识产权保护、民法、项目管理。
valentin.s.nikolaenko@tusur.ru

Статья поступила в редакцию 16.08.2024; после рецензирования 17.09.2024 принята к публикации 21.09.2024. Автор прочитал и одобрил окончательный вариант рукописи.
The article was submitted on 16.08.2024; revised on 17.09.2024 and accepted for publication on 21.09.2024. The author read and approved the final version of the manuscript.
文章于 16.08.2024 提交给编辑。文章于 17.09.2024 已审稿。之后于 21.09.2024 接受发表。作者已经阅读并批准了手稿的最终版本。