УДК 004

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ: КОМПЛЕКСНЫЙ ПОДХОД К ЭФФЕКТИВНОСТИ И КАЧЕСТВУ

Кряжева Е. В.

к.псих.н., доцент,

Калужский государственный университет им. К.Э. Циолковского,

Калуга, Россия

Ратников Р.Е.

студент,

Калужский государственный университет им. К.Э. Циолковского,

Калуга, Россия

Аннотация.

В статье рассматривается проблема разработки программного обеспечения с точки зрения обеспечения комплексного подхода к эффективности и качеству. Авторы анализируют этапы разработки программного обеспечения: от анализа требований до развертывания, освещаются ключевые методологии, лучшие практики и инструменты, которые способствуют успешным проектам разработки программного обеспечения. В конце авторами делаются выводы о перспективах такого подхода при разработке программного обеспечения.

Ключевые слова: программное обеспечение, комплексный подход, качество программного обеспечения, моделирование, метод приоритизации, тестирование, обслуживание, гарантия качества, управление проектом.

SOFTWARE DEVELOPMENT: AN INTEGRATED APPROACH TO EFFICIENCY AND QUALITY

Kryazheva E. V.,

Candidate of Psychological Sciences, Associate Professor,

Kaluga State University named after K.E. Tsiolkovsky,

Kaluga, Russia
Ratnikov R.E.,
student,
Kaluga State University named after K.E. Tsiolkovsky,
Kaluga, Russia

Annotation.
The article deals with the problem of software development from the point of view of providing an integrated approach to efficiency and quality. The authors analyze the stages of software development: from requirements analysis to deployment, highlighting key methodologies, best practices and tools that contribute to successful software development projects. In the end, the authors draw conclusions about the prospects of this approach in software development.
Keywords: software, integrated approach, software quality, modeling, prioritization method, testing, maintenance, quality assurance, project management.

Разработка программного обеспечения является важнейшим направлением развития современных информационных технологий; ее успех зависит от четко определенных процессов и методологий. В статье дается представление о ключевых компонентах разработки программного обеспечения, направленных на повышение эффективности и качества на протяжении всего жизненного цикла программного обеспечения [4]. Также разработка программного обеспечения требует комплексного подхода к эффективности и качеству, начиная с разработки соответствующей технической документации и заканчивая сопровождением готового программного продукта.

Под качеством программного обеспечения мы будем понимать степень соответствия функциональных, технических, эксплуатационных характеристик разработанного программного продукта целям, которые были поставлены перед началом его разработки. Разработку программного продукта можно представить

в виде пяти этапов: 1) проект; 2) состав проектной документации; 3) содержание проектной документации; 4) оформление проектной документации; 5) последовательность приемки системы.

Одним из ключевых аспектов комплексного подхода является анализ и планирование требований к программному продукту. Этот этап включает в себя сбор, анализ и документирование потребностей пользователей и системных требований, чтобы обеспечить четкое понимание желаемой функциональности и целей программного обеспечения.

Общие требования для оценки качества программной продукции рассматриваются в стандарте ГОСТ Р ИСО/МЭК 25040-2014 Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Процесс оценки. Анализ требований начинается с привлечения заинтересованных сторон, включая клиентов, конечных пользователей и экспертов в предметной области, для определения их ожиданий и целей. Такие методы, как интервью, опросы и семинары, способствуют сбору ценной информации. Затем эти требования документируются. Требования к содержанию документации регламентированы ГОСТ Р 59795–2021 [5].

Для дальнейшего уточнения и валидации требований разработчики используют различные инструменты и методологии. Моделирование вариантов использования позволяет визуализировать поведение системы с точки зрения пользователя, идентифицируя действующих лиц, варианты использования и их взаимодействия. Истории пользователей содержат краткие описания системных функций с точки зрения пользователя, уделяя особое внимание тому, "кто", "что" и "почему". Как только требования собраны, они подвергаются анализу для выявления потенциальных конфликтов, несоответствий или недостающей информации. Это помогает гарантировать, что требования являются полными, выполнимыми и соответствуют масштабу и целям проекта.

Методы приоритизации требований, такие как MoSCoW (Must have, Should have, Could have, Would have), помогают в определении критических функций,

которые необходимо предоставить. Планирование является неотъемлемой частью анализа требований, поскольку оно включает в себя определение объема проекта, установление контрольных точек, оценку ресурсов и установление реалистичных сроков. Руководители проекта и заинтересованные стороны сотрудничают для определения целей проекта, разбивая их на управляемые задачи и распределяя обязанности. Такие инструменты, как диаграммы Ганта или доски Канбана, помогают визуализировать временные рамки проекта и отслеживать прогресс. Эффективная коммуникация имеет важное значение во время анализа требований и планирования. Постоянное сотрудничество с заинтересованными сторонами гарантирует учет их меняющихся потребностей и ожиданий и надлежащее реагирование на любые изменения. Регулярные совещания о ходе работы, отчеты о проделанной работе и обзоры документации способствуют прозрачной и эффективной коммуникации.

Вкладывая время и усилия в тщательный анализ требований и планирование, команды разработчиков программного обеспечения могут заложить прочную основу для последующих этапов. Такое всестороннее понимание потребностей пользователей и системных требований позволяет разработчикам разрабатывать и внедрять программное обеспечение, которое соответствует ожиданиям, сокращает количество переделок и, в итоге, обеспечивает успешный программный продукт.

Следующим этапом комплексного подхода является проектирование и разработка дизайна и архитектуры, которые закладывают основу для разработки программного обеспечения. Этот этап включает в себя создание подробного проекта системы, в котором описываются структура, компоненты и взаимодействия программного обеспечения. Используя шаблоны проектирования, такие принципы, как модульность, и архитектурные стили, разработчики могут разрабатывать масштабируемые, поддерживаемые и расширяемые программные решения.

Далее идет реализация и кодирование. Этап внедрения включает в себя написание кода, основанного на дизайне и архитектуре. Разработчики должны придерживаться стандартов кодирования и лучших практик, чтобы обеспечить читаемость кода, ремонтопригодность и возможность повторного использования. Использование систем контроля версий и инструментов совместной разработки обеспечивает эффективную командную работу и эффективное управление кодом.

Следующий этап — это тестирование и гарантия качества. Тестирование имеет решающее значение для выявления и устранения дефектов в программном обеспечении [6]. Этот этап включает в себя различные методы тестирования, такие как модульное тестирование, интеграционное тестирование и системное тестирование. Платформы автоматизации тестирования и инструменты непрерывной интеграции повышают эффективность, обеспечивая более быстрые циклы обратной связи и раннее обнаружение ошибок. Применение методов обеспечения качества гарантирует поставку надежного и высококачественного программного обеспечения.

Далее - развертывание и обслуживание. Этап развертывания включает в себя упаковку программного обеспечения и развертывание его в производственной среде. Конвейеры непрерывной интеграции / непрерывного развертывания (CI/CD) автоматизируют процесс развертывания, снижая риск ошибок. После развертывания необходимо постоянное техническое обслуживание для устранения ошибок, добавления новых функций и оптимизации производительности, обеспечивая положительный пользовательский опыт.

Следующий этап: управление проектом и совместная работа. Эффективное управление проектом и совместная работа жизненно важны для успешной разработки программного обеспечения. Использование методологий управления проектами, таких как Agile или Scrum, облегчает итеративную разработку, способствует прозрачности и способствует эффективной коммуникации между

членами команды. Инструменты совместной работы, такие как системы отслеживания проблем и программное обеспечение для управления проектами, оптимизируют рабочие процессы и повышают производительность.

И на заключительной этапе комплексного подхода предлагается непрерывное совершенствование и DevOps [3]. Для команд разработчиков программного обеспечения постоянное совершенствование имеет решающее значение для улучшения своих процессов и предоставления более качественных продуктов. Внедрение методов DevOps, включая непрерывную интеграцию, непрерывную доставку и мониторинг, способствует формированию культуры совместной работы, автоматизации и разработки, основанной на обратной связи. Регулярные ретроспективы и сеансы обмена знаниями позволяют командам определять области для улучшения и оптимизировать жизненный цикл разработки программного обеспечения.

Таким образом, разработка программного обеспечения требует комплексного подхода, который охватывает различные этапы и дисциплины. Следуя лучшим практикам, используя соответствующие инструменты и постоянно совершенствуясь, разработчики могут создавать эффективные, высококачественные программные решения, которые отвечают потребностям пользователей и способствуют технологическому прогрессу.

Библиографический список:

1. Белый, Е.М. Управление проектами (с практикумом) / Е.М. Белый // М.: КноРус - 2019 – 262 с. ЭБС BOOK.ru https://www.book.ru/book/931302 (Дата обращения: 14.11.2022).

2. Гаврилова, Т.А. Инженерия знаний. Модели и методы [Электронный ресурс] : учеб. / Т.А. Гаврилова, Д.В. Кудрявцев, Д.И. Муромцев. // Электрон.дан. - Санкт-Петербург: Лань - 2016. – 324с.

3. Ганжур, М.А. Анализ методологий DEVOPS и DEVSECOPS / М.А. Ганжур, Н.В. Дьяченко, А.С. Отакулов // Молодой исследователь Дона. Донской государственный технический университет. – 2021. - №5 (21) – С.8-10

4. Гибкая методология разработки программного обеспечения: курс лекций/ - Москва: Интуит НОУ - 2016. - 154 с. URL:https://book.ru/book/917699 (Дата обращения: 08.11.2022).

5. Описание стандарта ГОСТ Р 50.1.028-2001 “Информационные технологии поддержки жизненного цикла продукции. Методология функционального моделирования” [Электронный ресурс] ИПК Издательство стандартов. - 42с. /URL: https://gostrf.com (дата обращения: 08.04.2023).

6. Стружков, С.А. Проблема формализации базовых показателей качества программного обеспечения на примере «сопровождаемости»/ С.А. Стружков // Труды Всероссийской научно-практической конференции «Транспорт России: проблемы и перспективы». М.: МИИТ - 2008 г. – С. 3-13.

Оригинальность 92%